April 2011

Sabertooth: A High Mobility Quadrupedal Robot Platform

Anton Zalutsky
Worcester Polytechnic Institute

Jeffrey Eugene O'Rourke
Worcester Polytechnic Institute

Jonathan Morgan
Worcester Polytechnic Institute

Timothy Flynn
Worcester Polytechnic Institute

Vadim Chernyak
Worcester Polytechnic Institute

Sabertooth Robotics

Vadim Chernyak (ME)
Timothy Flynn (RBE / CS)
Jonathan Morgan (RBE / ME)
Jeffrey O'Rourke (ME)
Anton Zalutsky (RBE)
Advisors:
Dr. Taskin Padir (RBE/ECE)
Dr. Sonia Chernova (RBE/CS)
Dr. Stephen S. Nestinger (RBE/ME)

Sabertooth@wpi.edu
http://www.sabertoothwpi.com

[SABERTOOTH: A HIGH MOBILITY QUADRUPEDAL ROBOT PLATFORM]

Sabertooth: A High Mobility Quadrupedal Robot Platform

A Major Qualifying Project
Submitted to the Faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfilment of the requirements for the
degree of Bachelor of Science

by
Vadim Chernyak

____
Timothy Flynn

____
Jonathan Morgan

____
Jeffrey O'Rourke

____
Anton Zalutsky

____
sabertooth@wpi.edu
http://www.sabertoothwpi.com

Date:
28 April 2011

Report Submitted to:

Professor Taskin Padir
Professor Sonia Chernova
Professor Stephen S. Nestinger

ABSTRACT

Team Sabertooth aimed to design and realize an innovative high mobility, quadrupedal robot platform capable of delivering a payload over terrain otherwise impassable by wheeled vehicles at a speed of 5 feet per second. Specifically, the robot is designed to be able to ascend and descend straight stairs with a predetermined gait. The robot uses a spring system in each of its legs for energy efficient locomotion. The 4ft x 3ft x 3ft freestanding four legged robot weighs approximately 300 pounds with an additional payload capacity of 30 pounds. An important feature of the robot is the passive, two degree of freedom body joint which allows flexibility in terms of robot motions for going around tight corners and ascending stairs. The sensor system integrates a LIDAR, an IMU and a camera for staircase recognition, obstacle avoidance, and distance calculation. A distributed control and software architecture is used for world mapping, path planning and motion control.

ACKNOWLEDGEMENTS

There are so many people that made this project possible and we wouldn't feel right going through this report without thanking everyone that helped along the way. First and foremost we'd like to thank all three of our advisors, Professor Taskin Padir, Professor Stephen S. Nestinger, and Professor Sonia Chernova. Without the help of these three diligent professors we would never have found the resources to complete this project, let alone graduate. We would especially like to all of our advisors for being able to help us troubleshoot all of our issues at any and all ungodly hours of the night.

Next we would like to say thank you to the two Independent Study students Paul Heslinga and Brennan Ashton. Whether it was yelling at each other about the communication protocol at 3 in the morning or deciding the best course of action for fixing an interference issue Paul and Brennan were there to help us solve the issue.

We would like to acknowledge all of our sponsors for all of the support that they have given us through the year. Without their belief that five undergraduate students could actually follow through on a project of this scale we would have never had the resources to do anything. Specifically, thank you to NI, Maxon, LakeShoreCarbide, SolidWorks, A123 Systems, QinetiQ, MathWorks, Yarde Metals, Venom, BobObots, Microstrain, Hydrocutter, Anderson Sign & Graphics, St. Pierre and Onvio. Thank you so much to all of you, your support really kept us going.

We would also like to thank all the Professors that have helped us on some portions of the project. Thank you to Professor James Van de Ven for his continual support helping us design a complex ankle system, Professor Eben Cobb for helping us with some of the more complex CAD work, Professor Robert Norton for helping us deal with those complex springs and cables and Professor Marco Popovic for his help with some of the more advanced modeling . We'd also like to acknowledge Ken Stafford, Brad Miller, and Joe St. Germain for helping us through some of the initial first stages and always peeking an interest in how the project is going.

Next, a huge thanks to our landlords, Toby, Adam and Neil. Thank you for giving us a home for the last 5 months and dealing with our mess and clear trash bags. Additionally thank you to our roommates James, JP and the rest of the FSAE team as well as Josh and the Bomb Disposal team. You've provided some fun times and some extreme distraction which likely kept us sane.

Moving on, we want to thank all the students who directly helped our project. Thank you to Andy Lewis, Brian Benson, Ciaran Murphy, Jake Dusek, George Hotz, Michael Fagan, Dan Jones, Colin Roddy, Rachel Heller, and Arthur Dutra for all their help on just about every element of the robot. It was really amazing knowing that we had so many students who would take time out of their already busy schedule to help us in time critical applications. Whether it was helping by tapping 2 holes or spending countless nights helping us go over our mechanical design thank you to all the students that made this project actually end somewhat on time.

Finally, the team would like to specifically acknowledge Professor Gennert for his belief in the team and continual support. Professor Gennert had faith in the team's ability to follow through on such a complex project and backed us the entire way. Without his belief the project would have never happened.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

EXECUTIVE SUMMARY

Sabertooth Robotics is a senior design team composed of five engineering students from the Worcester Polytechnic Institute. The team worked on an innovative robotics project whose scope was to design a high mobility, quadrupedal robot capable of traversing terrain otherwise impassable by wheeled vehicles. Specifically, the robot was designed to be able to ascend and descend straight stairs with a predetermined gait. With each step, the robot is designed to be able to recover energy exerted against the ground using a spring system.

Denavit-Hartenberg parameters were utilized to know where to place the IMU, LIDAR, and camera for understanding the perspective, position, and orientation in reference to the body for more accurate modeling. The IMU's placement was especially important to be placed as closely as possible to the actual revolute joint of the robot to properly analyze accelerations seen at the body joint. These parameters were then further used to begin modeling the forward and inverse kinematics of the system.

A simulation of Sabertooth was created in MATLAB because of its ability to effectively manipulate large quantities of data. This data was used to calculate motor requirements necessary to have Sabertooth walk at a desired speed of 5 ft/s, as well as gait generations that would show the ranges of motion that were required for the legs.

Utilization of the MATLAB simulation allowed for proper gait generation. By creating a gait loop which represents going through a gait from 0% to (final cycle+duty cycle)% where the duty cycle is represented by the max percentage of time the foot is spent off the ground and the final cycle is the percentage at which the last foot in the series leaves the ground. So as an example, in the bound gait the duty cycle is approximately 35% and the final cycle is approximately 70%- hence the loop will be from 0% to 105%. Lastly creating an offset for each leg is set as the first contact with the ground for the foot.

These simulations were used to create a mechanically robust system. In order to fully develop a complete mechanical system the team used SolidWorks for all of their necessary drafting needs. The team was able to work through multiple problems by carefully modeling every component in the system. SolidWorks proved to be an excellent modeling software from which we were able to develop full mechanical drawings. Upon completion of the full mechanical design, the team was able to separate the robotic system into different mechanical subassemblies.

The hip of the robot is the revolute joint perpendicular to the knee and shoulder joints that allow the legs to camber in and out from the body. This functionality is needed not only for foot placement on uneven terrain, but also for stabilization when the robot is off balance and for sidestepping and turning. The body joint needed to mimic a quadrupedal animal spine by having a 2 degree of freedom, highly compliant joint which would separate the front half of the body from the back half. The sensor head was designed to house the on-board camera, the LIDAR, and two arrays of 80-lumen LEDs.

One of the team's primary design requirements included implementing the SLIP model into our design. The team grasped that the most effective way to implement the SLIP model would be to introduce springs in series with the motor. This design would need to encompass capstons for the motors, springs in line with the motor, and capstons for both the shoulder and knee system. Additionally, the design called for an idler pulley at the shoulder to drive the knee. This was necessary so that when the shoulder was moving, the effective length to the knee was not affected. The entire cable system was also prototyped in full scale to verify both the spring system and to check for interferences in such a system.

The anatomy of a German Shepard was examined to isolate the unique traits that are seen in both the front and rear legs. Each of these leg systems is designed to serve a particular purpose and the design requirements were aimed to be replicated in the mechanics of the legs designed for the robot. The front paws and ankles are designed with a particular motif in mind. While the rear ankles are designed to be the main power supply for the gait the front ankles are designed for finesse and precise motion. Each of these leg designs provide delicate design characteristics which were followed during the robots design process.

The link between the software and mechanical subsystems was accomplished through the use of a National Instruments SBRIO. In an effort to properly interface with the National Instruments SBRIO both mechanically and through software, the team developed a PCB. The PCB was designed mainly as a breakout board, but also incorporated operational amplifiers for sensor inputs.

The power requirements for this system are unique and range from 28 V to 5V; this difference in voltage requires the use of a DC-DC converter which can properly regulate voltage differences. In order to provide battery power for this system, the team chose to go with 96 A123 26650 cells. These voltages served to be the largest causes of noise in the system. In addition, the high current requirements of the robot require the emergency stops which were connected to a relay bank in order to properly stop motion of the motors.

There are two different motor requirements on the robot, these requirements were met through the use of both the Maxon EC40Max and Ampflow A28-150 motors. These motors were controlled through two different motor controllers. The first controller that was explored was the Maxon EPOS2 70/10, which was the controller that was the Maxon controller properly paired with the ECMax. The second controller was the RoboteQ single channel motor controller. This motor controller is able to supply one DC brushed motor with 120A continuous with the ability to peak at 500A for a few seconds.

The control is accomplished through the use of the sbRIO 9632 which integrates a real-time processor, a field programmable array, and I/O on one board. The real-time processer is in charge of receiving packets of data from the netbook with motor positions every 50ms through UDP packets. Acting as a medium, the processor passes information from the computer to the FPGA. As this data streams to the processor, it computes how to handle this information based on several conditional statements regarding current, and position control.

There is the main interface with Sabertooth allowing for PID tuning, velocity profiling, and full system monitoring. Displays stating the velocity being sent to each motor controller, every potentiometer input, and current through each motor controller are shown on this screen. A large software Emergency Stop button allows for each motor to be halted while communication with the system is available and doesn't require a full system reboot if triggered.

The complete software architecture is divided into a two-process, multi-threaded application. The first process is used for all on-board control. It starts with the executive, which spawns off multiple threads for the other blocks involved in that process (the planner, knowledge base, sensors, and pilot/actuators). The second process is a Web-based user interface used to interact with a remote user.

The user interface (UI) we developed serves two purposes. First, it allows outside users to view the robot and monitor what it is doing and planning to do. The viewable data includes a live feed of the on-board camera, angular position data for each link, and Cartesian coordinates of where the robot is (relative to its starting position) as well as where it plans on going. Second, the UI allows for those with correct login information to control the robot.

The design process was brought to completion with the assembly and testing of the entire robot. Each of the development systems, mechanical, electrical and software, were realized and design problems were identified. Through this process many small design flaws were found and fixed in order to create a working prototype.

CHAPTER 1: INTRODUCTION

Current estimates show that 70% of the earth's landmass is not accessible to wheeled or tracked machines. Human beings, animals and robotic systems with legs, however, are able to navigate through these terrains, stepping over obstacles and exploring new ground. These systems are able to comply with the challenging terrain, stepping into ditches and over obstacles. The past 40 years have engaged a number of mathematicians, engineers and biologists to study legged locomotion and how we can conquer the terrain presented both by nature and complex man made obstacles. One approach to further understanding legged motion has been to study animals which have been optimized through millions of years of evolution. From these animals, mathematical models have been developed to further understand the optimized gaits the employ to conquer this challenging terrain (Hilderbrand, 1965). These models include accurate foot placement, bone lengths, and associated spring forces to accurately model how animals are able to change their gaits in a dynamic fashion.

While human beings and animals have found effective and efficient ways to navigate this complex terrain, modern day robotic systems are still extremely limited in their capabilities. There has been very minimal research in the field of legged robotics with only a handful of robots actually being able to traverse through the earth's varying terrains (Raibert, et al., 2008). Even modern day systems which are able to navigate this terrain suffer through their own respective drawbacks including energy inefficiency, noisy operation and the inability to change their gait algorithm to interact with the constantly varying terrain.

If a robotic system was developed to overcome these challenges and current limitations then earth's landmass could be further explored. This task becomes exceedingly important in crossing potentially dangerous terrain which humans are not capable of passing without extreme danger. A robotic system which is able to efficiently and effectively cross this path would be invaluable in situations such as these.

The ultimate goal of this project is to develop an innovative robotics project whose scope is to design a high mobility quadruped robot capable of traversing terrain otherwise impassable by wheeled vehicles. The main innovation of the robot originates from the mechanical compliance aspect of the system through the ankle, spring, and body joint subsystems. With each step, the robot will be able to recover energy lost during that step using a spring system.

This project was completed by a series of students whose interests range over multiple interdisciplinary professions. The students and their respective leads in the project are displayed in Table 1 in the Responsibility Assignment Matrix.

Task	Responsible (R)	Assists (A)	Consulted (C)	Informed (I)
Hip Subsystem (Mechanical)	Jeffrey O’Rourke	Vadim Chernyak	Jonathan Morgan	Anton Zalutsky
Body Joint Subsystem (Mechanical)	Vadim Chernyak	Jeffrey O’Rourke	Jonathan Morgan	Timothy Flynn
Sensor Head Subsystem (Mechanical)	Jonathan Morgan	Vadim Chernyak	Timothy Flynn	Jeffrey O’Rourke
Body Frame Design (Mechanical)	Jeffrey O’Rourke	Jonathan Morgan	Vadim Chernyak	Anton Zalutsky
Leg/Cable/Spring Design Subsystem	Vadim Chernyak	Jonathan Morgan	Jeffrey O’Rourke	Anton Zalutsky
Ankle Design	Jonathan Morgan	Jeffrey O’Rourke	Vadim Chernyak	Anton Zalutsky
Manufacturing Logistics	Vadim Chernyak	Jonathan Morgan	Jeffrey O’Rourke	Anton Zalutsky
MATLAB Simulations	Anton Zalutsky	Vadim Chernyak	Timothy Flynn	Jonathan Morgan
Sensor Head Integration	Timothy Flynn	Anton Zalutsky	Jonathan Morgan	Vadim Chernyak
SBRio Board Integration	Anton Zalutsky	Timothy Flynn	Jonathan Morgan	Jeffrey O’Rourke
ROS/Rviz Integration	Timothy Flynn	Anton Zalutsky	Vadim Chernyak	Jonathan Morgan

Table 1 – Responsibility Matrix

The tasks outlined in the responsibility matrix were mostly completed. The robot has been completed to the point where the robot is capable of traversing through a gait on a test harness. Therefore, it has not yet reached the point where the integration of sensors responsible for gathering data about the environment is not yet utilized. However, the software for those sensors has been completely developed.

CHAPTER 2: BACKGROUND RESEARCH

In order to fully understand both what has been accomplished in the field of legged locomotion as well as current limitations, past research in this field was analyzed and evaluated. Additionally, research in current technology required for this field was also conducted. This research included vision systems, sensor technology, power transmission and cam design.

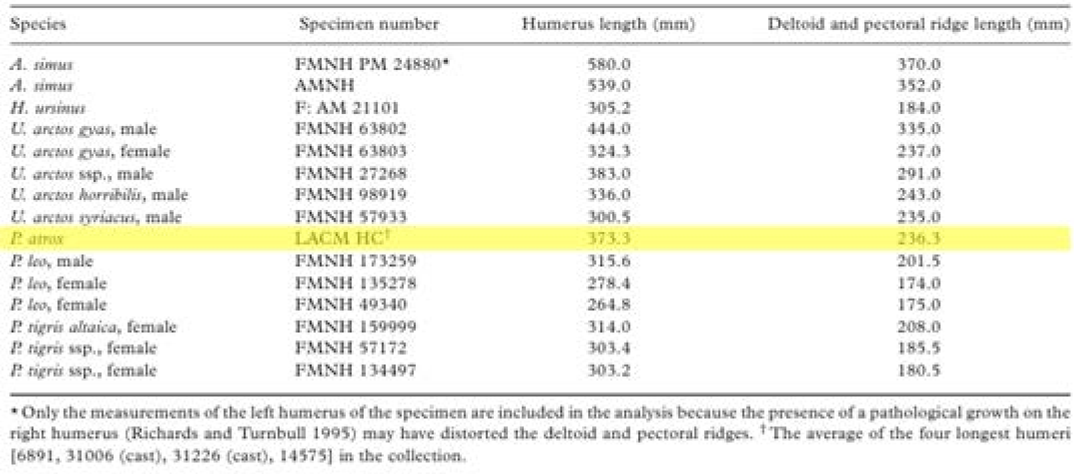
2.0 Animal Locomotion

Before starting any of the detailed mechanical research it was necessary to understand how nature takes on the complicated task of quadrupedal motion. This research led to a focus on the anatomy and locomotion of dogs, cats and horses. All of these quadrupeds interact with the environment differently and by exploring their locomotion we were able to understand some of the reasoning behind their anatomy and the gaits that they go through.

2.0.1 Anatomy

Animal anatomy plays a very large role in the inspiration for this project. It is necessary to carefully note animal sizing and its impact on animals capability of performing specific gaits as well as interacting with varying terrain. While there are many aspects of anatomy that could contribute to quadruped interaction with the environment, it was decided to focus specifically on the sizing of the legs, ratio of ankle movement, ankle tendons, and the spine. The research into each of these systems directly impacted mechanical design and the analysis of how well each system performed.

The anatomy of quadrupeds was explored and a focus on the lengths of the bones in the legs, specifically the humerus, ulna, femur, tibia, metacarpals and metatarsals was examined. These lengths were calculated using the data and ratios obtained from the article *Morphological Comparison of* *big Pantherines and brown bears.* The lengths of these bones were realized by using the following tables. The length of the Humerus was determined through the use of Table 2. It was determined that a link length around 14-16 inch would be the ideal size for the quadruped. This length was equal to roughly 330-400mm leading to the selection of the species *Panthera atrox* with a link length of 373.3mm.



Species	Specimen number	Humerus length (mm)	Deltoid and pectoral ridge length (mm)
A. simus	FMNH PM 24880*	580.0	370.0
A. simus	AMNH	539.0	352.0
H. ursinus	F: AM 21101	305.2	184.0
U. arctos gyas, male	FMNH 63802	444.0	335.0
U. arctos gyas, female	FMNH 63803	324.3	237.0
U. arctos ssp., male	FMNH 27268	383.0	291.0
U. arctos horribilis, male	FMNH 98919	336.0	243.0
U. arctos syriacus, male	FMNH 57933	300.5	235.0
P. atrox	LACM HC†	373.3	236.3
P. leo, male	FMNH 173259	315.6	201.5
P. leo, female	FMNH 135278	278.4	174.0
P. leo, female	FMNH 49340	264.8	175.0
P. tigris altaica, female	FMNH 159999	314.0	208.0
P. tigris ssp., female	FMNH 57172	303.4	185.5
P. tigris ssp., female	FMNH 134497	303.2	180.5

* Only the measurements of the left humerus of the specimen are included in the analysis because the presence of a pathological growth on the right humerus (Richards and Turnbull 1995) may have distorted the deltoid and pectoral ridges. † The average of the four longest humeri [6891, 31006 (cast), 31226 (cast), 14575] in the collection.

Table 2 - Deltoid and Pectorialis leverage in Ursidae and Big Cats

The species *Panthera atrox* correlates to the American lion, "The American lion closely resembled the modern African lion, but was about 25% larger and heavier than the living species." (Tom Demere , 2010). A picture of the Panthera Leo Atrox can be found in Figure 1.

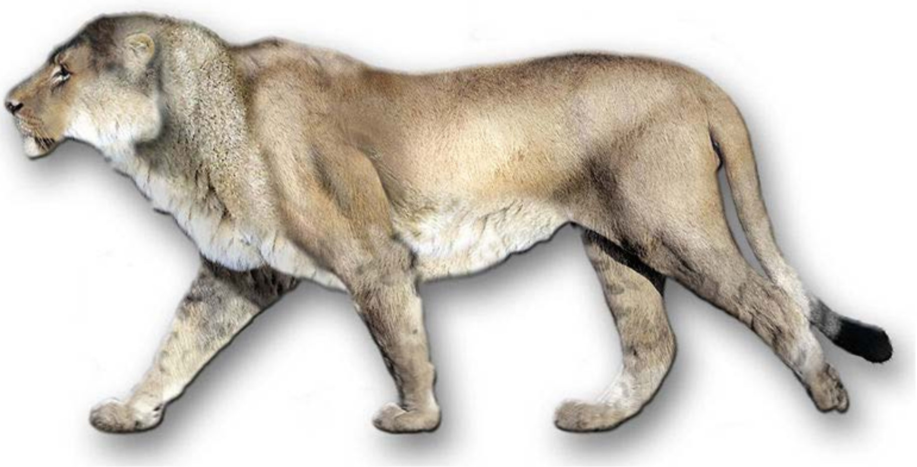
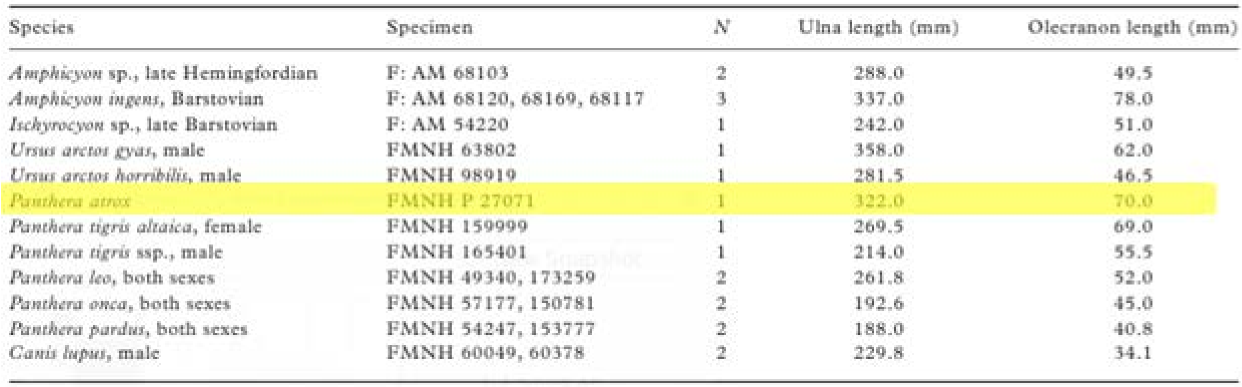


Figure 1 - Panthera Leo Atrox

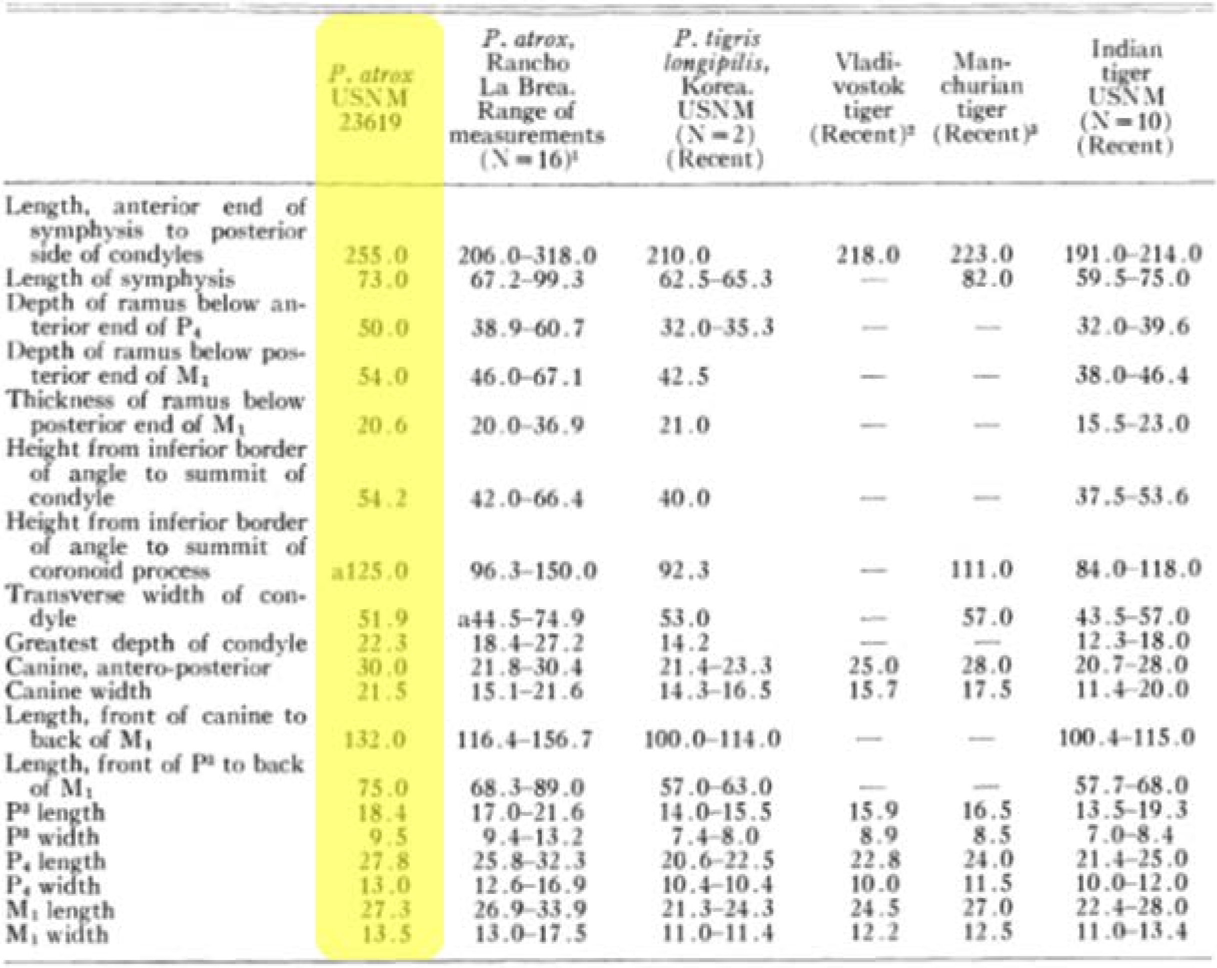
The length of the ulna was determined through the use of Table 3.



Species	Specimen	*N*	Ulna length (mm)	Olecranon length (mm)
Amphicyon sp., late Hemingfordian	F: AM 68103	2	288.0	49.5
Amphicyon ingens, Barstovian	F: AM 68120, 68169, 68117	3	337.0	78.0
Ischyrocyon sp., late Barstovian	F: AM 54220	1	242.0	51.0
Ursus arctos gyas, male	FMNH 63802	1	358.0	62.0
Ursus arctos horribilis, male	FMNH 98919	1	281.5	46.5
Panthera atrox	FMNH P 27071	1	322.0	70.0
Panthera tigris altaica, female	FMNH 159999	1	269.5	69.0
Panthera tigris ssp., male	FMNH 165401	1	214.0	55.5
Panthera leo, both sexes	FMNH 49340, 173259	2	261.8	52.0
Panthera onca, both sexes	FMNH 57177, 150781	2	192.6	45.0
Panthera pardus, both sexes	FMNH 54247, 153777	2	188.0	40.8
Canis lupus, male	FMNH 60049, 60378	2	229.8	34.1

Table 3- Triceps leverage in Amphicyon and Ischyrocyon and other Carnivora

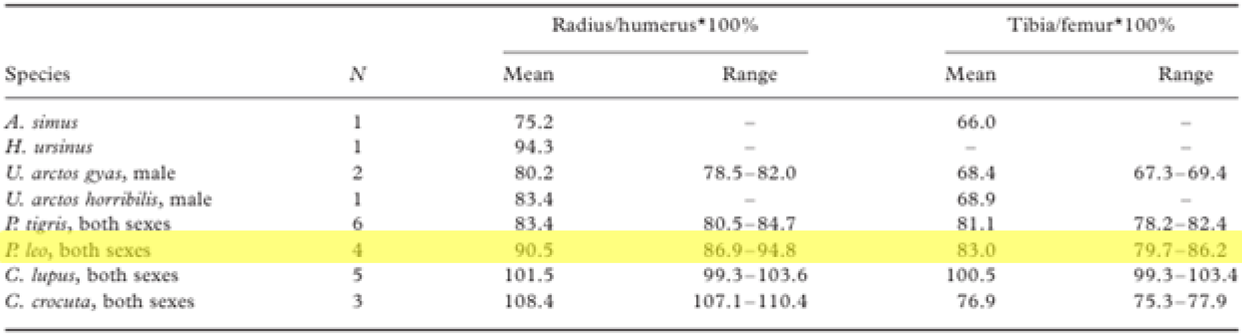
An ulna length of 322mm could be determined based off *Panthera Atrox* anatomy. The length of the femur was determined based on Table 4 and the corresponding P length value. This value was calculated to be roughly 16-18 in.



	P. atrox USNM 23619	*P. atrox*, Rancho La Brea. Range of measurements (N = 16)[1]	*P. tigris longipilis*, Korea. USNM (N = 2) (Recent)	Vladivostok tiger (Recent)[2]	Manchurian tiger (Recent)[3]	Indian tiger USNM (N = 10) (Recent)
Length, anterior end of symphysis to posterior side of condyles	255.0	206.0–318.0	210.0	218.0	223.0	191.0–214.0
Length of symphysis	73.0	67.2–99.3	62.5–65.3	—	82.0	59.5–75.0
Depth of ramus below anterior end of P_4	50.0	38.9–60.7	32.0–35.3	—	—	32.0–39.6
Depth of ramus below posterior end of M_1	54.0	46.0–67.1	42.5	—	—	38.0–46.4
Thickness of ramus below posterior end of M_1	20.6	20.0–36.9	21.0	—	—	15.5–23.0
Height from inferior border of angle to summit of condyle	54.2	42.0–66.4	40.0	—	—	37.5–53.6
Height from inferior border of angle to summit of coronoid process	a125.0	96.3–150.0	92.3	—	111.0	84.0–118.0
Transverse width of condyle	51.9	a44.5–74.9	53.0	—	57.0	43.5–57.0
Greatest depth of condyle	22.3	18.4–27.2	14.2	—	—	12.3–18.0
Canine, antero-posterior	30.0	21.8–30.4	21.4–23.3	25.0	28.0	20.7–28.0
Canine width	21.5	15.1–21.6	14.3–16.5	15.7	17.5	11.4–20.0
Length, front of canine to back of M_1	132.0	116.4–156.7	100.0–114.0	—	—	100.4–115.0
Length, front of P^3 to back of M_1	75.0	68.3–89.0	57.0–63.0	—	—	57.7–68.0
P^3 length	18.4	17.0–21.6	14.0–15.5	15.9	16.5	13.5–19.3
P^3 width	9.5	9.4–13.2	7.4–8.0	8.9	8.5	7.0–8.4
P_4 length	27.8	25.8–32.3	20.6–22.5	22.8	24.0	21.4–25.0
P_4 width	13.0	12.6–16.9	10.4–10.4	10.0	11.5	10.0–12.0
M_1 length	27.3	26.9–33.9	21.3–24.3	24.5	27.0	22.4–28.0
M_1 width	13.5	13.0–17.5	11.0–11.4	12.2	12.5	11.0–13.4

Table 4 - Measurements of Mandibles of Panthera Atrox and Panthera Tigirs

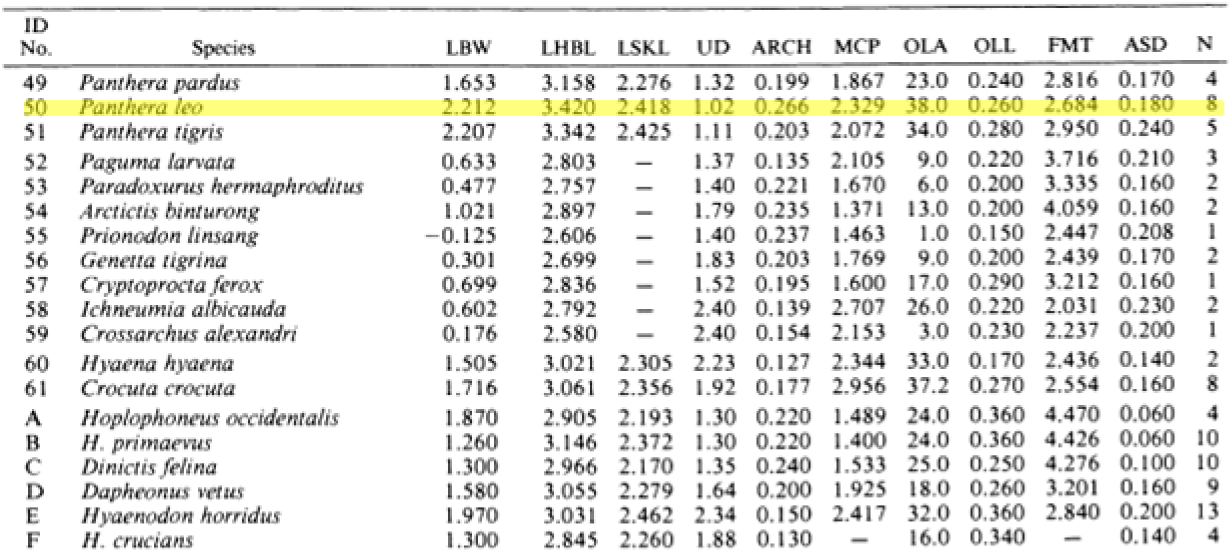
The bone length of the tibia was calculated based on the tibia/femur ratio found in Table 5. This ratio was based on *Panthera Leo* as upposed to *Pathera atrox* because of the species similarities in physical shape. This ration was found to be around 90% for the tibia to the femur.



		Radius/humerus*100%		Tibia/femur*100%	
Species	*N*	Mean	Range	Mean	Range
A. simus	1	75.2	–	66.0	–
H. ursinus	1	94.3	–	–	–
U. arctos gyas, male	2	80.2	78.5–82.0	68.4	67.3–69.4
U. arctos horribilis, male	1	83.4	–	68.9	–
P. tigris, both sexes	6	83.4	80.5–84.7	81.1	78.2–82.4
P. leo, both sexes	4	90.5	86.9–94.8	83.0	79.7–86.2
C. lupus, both sexes	5	101.5	99.3–103.6	100.5	99.3–103.4
C. crocuta, both sexes	3	108.4	107.1–110.4	76.9	75.3–77.9

Table 5 - Relative limb segment lengths

Using the length of the femur in conjunction with Table 6 the length of the metatarsals were able to be found, the FMT column refers to femur/metatarsal length. Using the ratio of 2.684 it was determined that the metatarsals should be approximately 5.5-6 in.



ID No.	Species	LBW	LHBL	LSKL	UD	ARCH	MCP	OLA	OLL	FMT	ASD	N
49	*Panthera pardus*	1.653	3.158	2.276	1.32	0.199	1.867	23.0	0.240	2.816	0.170	4
50	*Panthera leo*	2.212	3.420	2.418	1.02	0.266	2.329	38.0	0.260	2.684	0.180	8
51	*Panthera tigris*	2.207	3.342	2.425	1.11	0.203	2.072	34.0	0.280	2.950	0.240	5
52	*Paguma larvata*	0.633	2.803	–	1.37	0.135	2.105	9.0	0.220	3.716	0.210	3
53	*Paradoxurus hermaphroditus*	0.477	2.757	–	1.40	0.221	1.670	6.0	0.200	3.335	0.160	2
54	*Arctictis binturong*	1.021	2.897	–	1.79	0.235	1.371	13.0	0.200	4.059	0.160	2
55	*Prionodon linsang*	−0.125	2.606	–	1.40	0.237	1.463	1.0	0.150	2.447	0.208	1
56	*Genetta tigrina*	0.301	2.699	–	1.83	0.203	1.769	9.0	0.200	2.439	0.170	2
57	*Cryptoprocta ferox*	0.699	2.836	–	1.52	0.195	1.600	17.0	0.290	3.212	0.160	1
58	*Ichneumia albicauda*	0.602	2.792	–	2.40	0.139	2.707	26.0	0.220	2.031	0.230	2
59	*Crossarchus alexandri*	0.176	2.580	–	2.40	0.154	2.153	3.0	0.230	2.237	0.200	1
60	*Hyaena hyaena*	1.505	3.021	2.305	2.23	0.127	2.344	33.0	0.170	2.436	0.140	2
61	*Crocuta crocuta*	1.716	3.061	2.356	1.92	0.177	2.956	37.2	0.270	2.554	0.160	8
A	*Hoplophoneus occidentalis*	1.870	2.905	2.193	1.30	0.220	1.489	24.0	0.360	4.470	0.060	4
B	*H. primaevus*	1.260	3.146	2.372	1.30	0.220	1.400	24.0	0.360	4.426	0.060	10
C	*Dinictis felina*	1.300	2.966	2.170	1.35	0.240	1.533	25.0	0.250	4.276	0.100	10
D	*Dapheonus vetus*	1.580	3.055	2.279	1.64	0.200	1.925	18.0	0.260	3.201	0.160	9
E	*Hyaenodon horridus*	1.970	3.031	2.462	2.34	0.150	2.417	32.0	0.360	2.840	0.200	13
F	*H. crucians*	1.300	2.845	2.260	1.88	0.130	–	16.0	0.340	–	0.140	4

Table 6 - Skeletal Indicators of Locomotor Behavior

2.0.2 Ankle Design

The design requirements for this quadruped included the ability to explore further dynamic gaits as a research platform. This requirement highly influenced many design decisions and shaped the focus of the project into biomimetic research. Biomimetic research lays its foundation in creating systems that serve to imitate nature. By utilizing these systems we can further learn more about the mechanisms they are mimicking.

The anatomy of a German Shepard was examined to isolate the unique traits that are seen in both the front and rear legs. Each of these leg systems is designed to serve a purpose and their functionality was replicated in the mechanics of the legs designed for the robot. The front paws and ankles are designed with a particular motif in mind. While the rear ankles are designed to be the main power supply for the gait the front ankles are designed for finesse and precise motion. Each of these leg designs provide delicate design characteristics which were followed during the robots design process.

The front leg of the robot was based off of the paw and ankle movement in German shepherd dog forequarters seen in Figure 2. The front feet are required to withstand the maximum body weight impact compared to the impacts seen by the rear feet. Both the front and rear ankle systems consists of the bones of the feet as well as the paw, in the case of the front foot the paw is substantially larger than that of the rear to account for the increased forces it will experience.

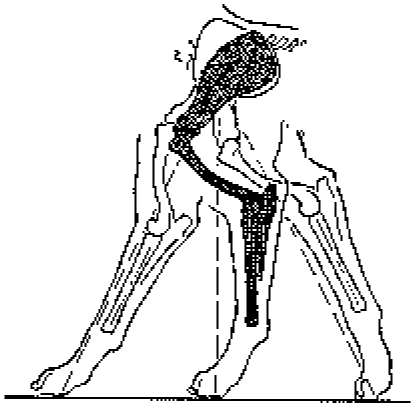


Figure 2 - German Shepard Forequarter

The hindquarter of the German shephard is the main method of power transfer between the dog and the ground. This power transfer is accomplished through the precise anatomy and angulations of the bones in the rear leg and shoulder. Any weakness in the antomical structure of the hindquarters will serve to compromise the ability to perform proper gaits. The basic anatomy of the hindquarter can be seen in Figure 3.

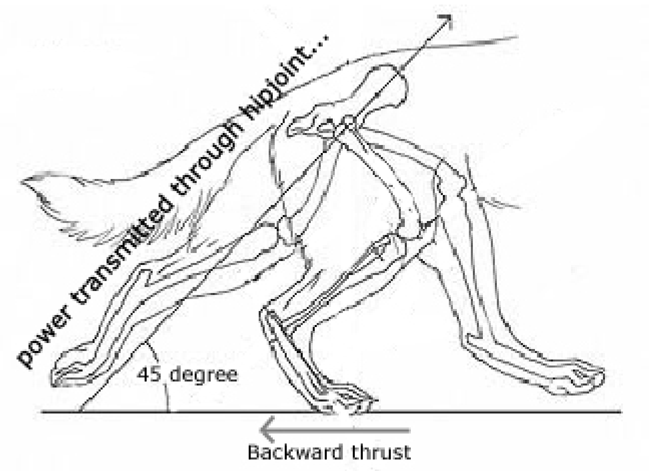


Figure 3 - German Shepard Heinquarter

Figure 3 depicts the powerful push off stride that is responsible for transferring the forces through the paw to the spine. This powerful gait is responsible for carrying most of the animal's mass through the gait, allowing for the forequarter to catch the mass and move with precision to the next position of power. Once the rear leg has finished its power push, the paw will lift off of the ground the metacarpals will form a 45 degree angle with the ground. This angle with the ground is controlled by the anatomical structure of the hindquarters and allows for proper placement of the paw throughout both the push and return strides of a particular gait.

Exploring the proper anatomical structure of the hindquarter will ensure the correct motion of the robot's legs and therefore optimize the amount of force which can be transferred to provide forward motion. This anatomical system consists of three main subsystems: the physical bones, the muscle control of the leg, and the tendons connecting the joints of the leg.

Two of these systems have been accounted for in the current leg design and the introduction of the front and rear ankle system will complete the system by recreating the tendons connecting the joints of the leg. The tendons located within the legs of quadrupeds are able to aid in the power regeneration and compliance aspects of the legs. These quadrupeds have very similar tendon anatomy, but all require the tendons to have varying non-linear spring forces in order to properly interact with the other leg compliances of that given animal. For the purposes of explaining this system, a bipedal robot will be explored referencing research from *Achilles: A robot with realistic legs* (Lewis and Klein, 2009). The Achilles robot utilizes biarticulate muscles to create an accurate representation of the human leg. The layout seen in Figure 4 is a cutaway view of the robotic legs intricate system. The muscles and tendons in the leg are labeled to their appropriate actuators. The biarticulate actuators which are important to this robotic system can be seen as the gastrocnemius and the rectus femoris.

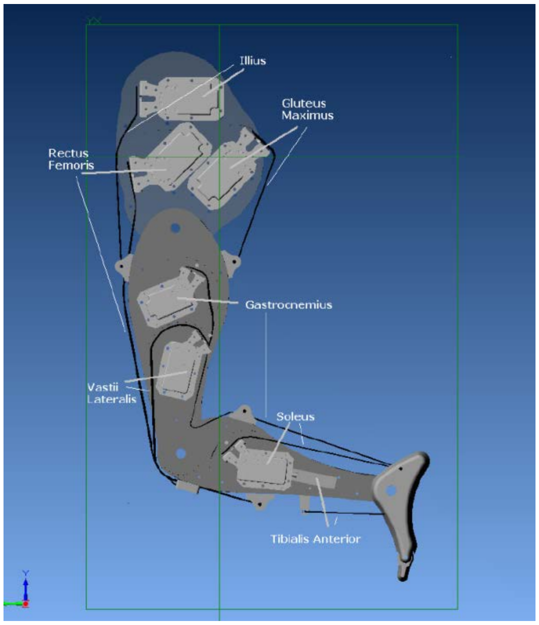


Figure 4 - Achilles Robot (Lewis and Klein, 2009)

The biarticulate muscles that can be modeled in the Achilles system as well as the soleus muscle play a critical role in the tendon recreation of the front and rear ankle systems. Both the front and rear ankle system work to model the anatomic aspects of this third tendon system. The benefits of this system focus on the power output that the tendons can provide to the system. While the front and rear ankle systems each serve the distinct purpose of finesse and power respectively, there is a necessity for each to have a level of power output. The front paws' ability to handle the majority of the animals weight through the impact force of the walking gait can be partially accounted for through these biarticulate tendons. On the other hand, the ability of the rear ankle to provide the necessary pushing power through each stride will be accounted for both through the muscle and tendon systems. The Achilles robotic system was able to provide the test data in Figure 5 regarding the amount of power each of these tendons would provide both together and individually.

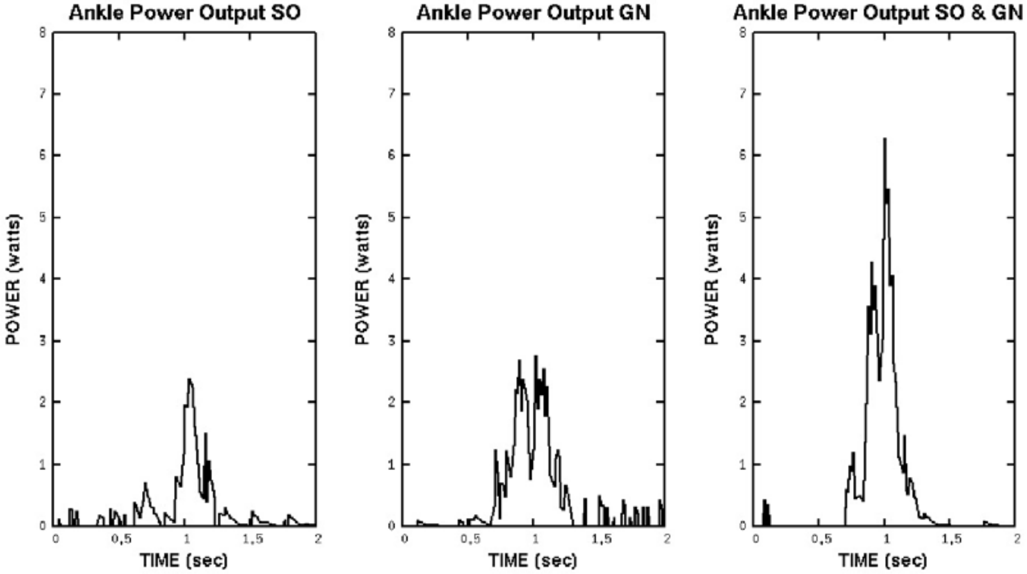


Figure 5 - Power versus time at ankle during return from a squat: (A) SO alone, (B) GA alone (C) SO and GA (Lewis and Klein, 2009)

As seen in Figure 5, the power output of the soleus and the gastrocnemius are comparable for peak outputs, but differ greatly when it comes to the length over which this power output can be maintained. In order to properly adapt these two muscles to the differing anatomy of the front and back legs the design requirements needed to be clarified. With the performance requirements of both the front and back ankles realized, Table 7 was created in order to relate these performances to power output and duration.

Ankle System	Power Output	Power Duration
Front	High Peak	Short Burst
Rear	High Power	Long Duration

Table 7 - Ankle Power Requirements

The next step was to analyze the feet that would be implemented as the robot's final end effectors. The feet needed to ensure the biological aspects of a multi-jointed paw was followed in order to recreate the flow of a normal gait. The gait of bipeds as well as quadrapeds are able to reproduce an efficient walking gait through through careful ground contact paired with proper link control. Bipeds are able to create a very efficient walking gait by employing the rolling foot model seen in Figure 6.

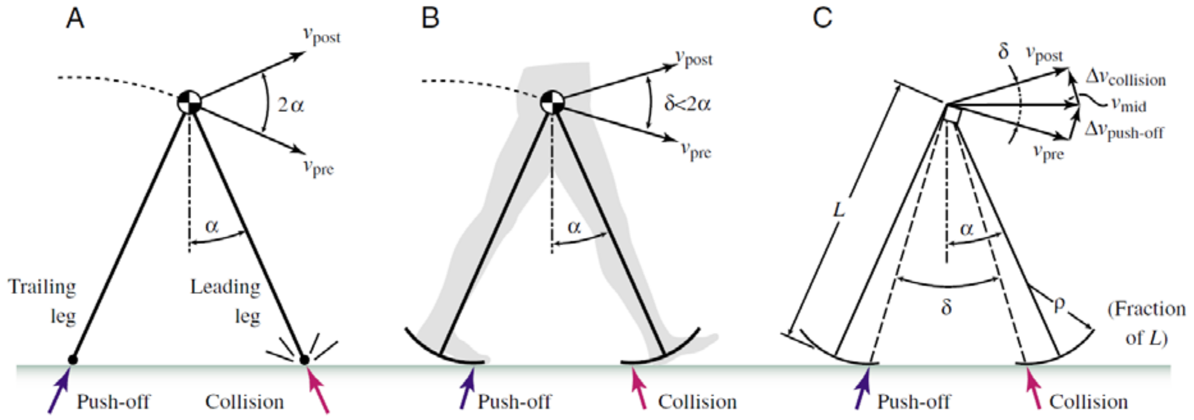


Figure 6 - Rolling Foot Model (Adamczyk, 2006)

The rolling foot model is based highly on the fluidity of a step in a constant velocity gait. Each step consists of three unique stages: collision, transition and push-off. The biped feet are able to successfully transfer the downward force into forward motion. They are able to effectively roll through the three stages: engaging the collision with the heel of a foot, transitioning through the arc of the of foot, and pushing off with the toes. There have been kinematic biped walkers which are able to accomplish consistent fluid forward motion by implementing a simple two link leg in conjunction with a curved foot that allows for the proper transmission of energy. Of these systems the McGeer walker at Cornell University has been one of the most successful examples (Garcia, Chatterjee, Ruina, 1999). The McGeer walker was able to maintain a consistent walking pace down an incline simply by transferring the potential energy through a linkage system. One of the McGeer walkers can be seen in Figure 7. This figure shows the full transition through a step using time lapse photography and a strobe flash.

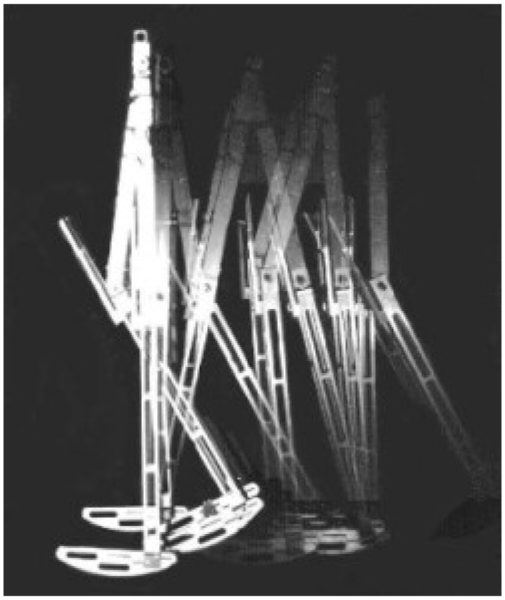
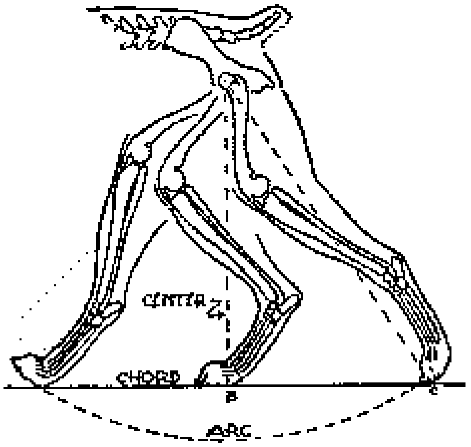


Figure 7 - Robotic Biped (Garcia, Chatterjee, Ruina, 1999)

Just as with the Spring Loaded Inverted Pendulum (SLIP) model (further explained in Section 2.5), this fluid stepping motion can also be applied to other walkers in nature. As seen in Figure 8 the motion of a quadruped step also creates a motion curve within the foot. However, with a quadruped the step starts on the toes and rolls back to the heels before the foot is lifted off of the ground to go through the idle motion of the gait (Cohen, 2003). Both the front and rear legs exhibit this behavior, only altering their total amount of contact time and the size of the footprint interacting with the ground plane.



Actually the pad remains fixed and the pelvis is carried forward, but figuratively the action is along a chord of an arc described by the leg action. During the first half of the chord, the leg receives concussion, but from B to C it is transmitting power generated by the rearing muscles. Efficiency is in ratio to the difference between AB and AC, the contracted and extended lengths.

Figure 8 - Foot Curve Quadruped (Oricom Technologies, 2001)

The curves generated by bipedal motion is determinable based on link lengths, but those curves corresponding to quadrupedal motion depend highly on not only the animal but the specific gait the animal is maintaining. These complex curves are generated due to the magnitude of bones and cartilage that are contained within the metacarpals and metatarsals of quadrapeds (Cohen, 2003). Fortunately as with other aspects of this robot, there was an aim to simplify the motion of the foot to its simpliest fluid motion. This was accomplished through careful examination of foot movement through a gait.

This examination allowed a pattern to emerge in regards to motion of the ankle and paw. Irregardless of the gait through which the quadruped was going through, the paw performed an almost identical motion. The only thing that changed based on the gaits was the frequency. This information was then used to further understand the ground contact of the foot for both the front and rear ankles.

Both the front and rear ankle systems were analyzed seperately. First, a graph of a leg going through a specific gait motion was obtained and examined. This graph seen in Figure 9 allowed for the creation of a curve of ground contact for the front ankle.

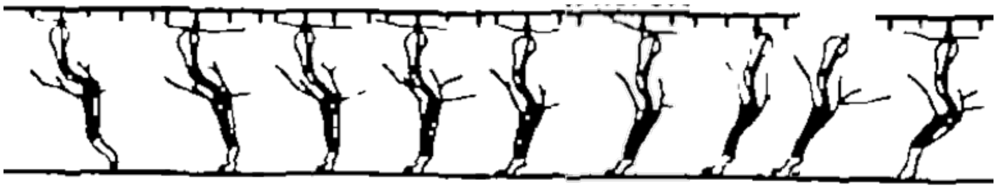


Figure 9 - Front Leg Gait Motion (Goslow, 1980)

The gait motion graph allowed for the position of the upper leg, lower leg and foot to be fully analyzed and applied to the fluidity of the system. The concept of fluid motion was continued with the creation of Figure 10 which took the union of all ground contact from Figure 9 and fit a curve for future use.

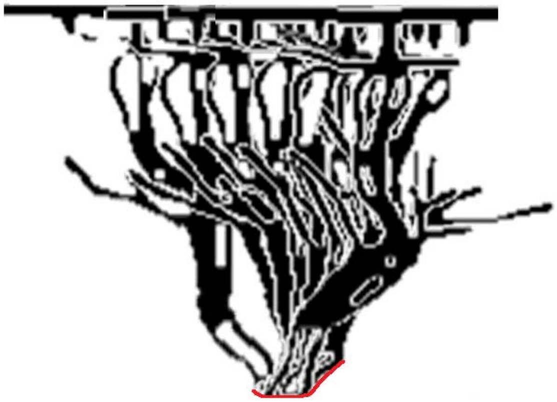


Figure 10 - Front Foot Curve Generation,
The curve of the front foot was calculated by smoothing out the curve created by all of the motion of the front foot. The gait motion timeline is split into motion steps and these motion steps are placed in a layer format creating the image above. This image then has a curve drawn (red) to represent the motion of the foot through the gait.

With a successful generation of a curve for the front foot, it was necessary to create a curve for the rear foot as well. This graph seen in Figure 11 allowed for the creation of a curve of ground contact for the front ankle.

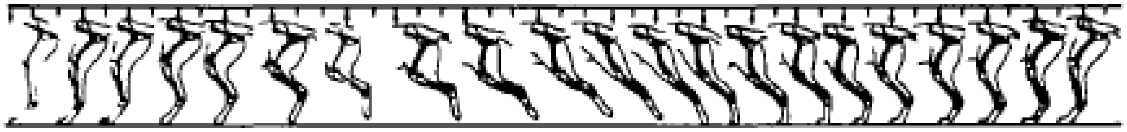


Figure 11 - Rear Leg Gait Motion (Goslow, 1980)

The concept of fluid motion was continued with the creation of Figure 12 which took the union of all ground contact from Figure 11 and fit a curve for future use.

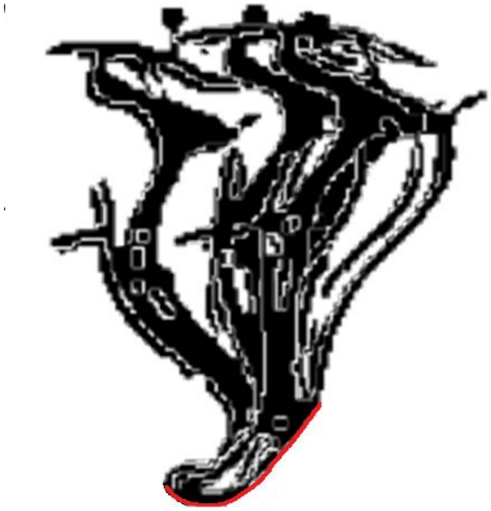


Figure 12 - Rear Foot Curve Generation
The curve of the rear foot was calculated by smoothing out the curve created by all of the motion of the rear foot. The gait motion timeline is split into motion steps and these motion steps are placed in a layer format creating the image above. This image then has a curve drawn to represent the motion of the foot through the gait.

2.0.3 Gaits

A gait is a pattern of motion of the limbs that allows for movement on a terrain. There are various gaits that can be used for various situations depending on speed, terrain, dexterity, and efficiency. A gait is typically described as a pattern regarding footfall, the foot's contact with the ground. A gait is classified in two forms, symmetrical and asymmetrical based on the movements of the limbs. The term symmetry does not imply that the left and right limbs are moving through the same motion. Instead, in a symmetrical gait, the left and right limbs alternate motion. In an asymmetrical gait, the limbs move together.

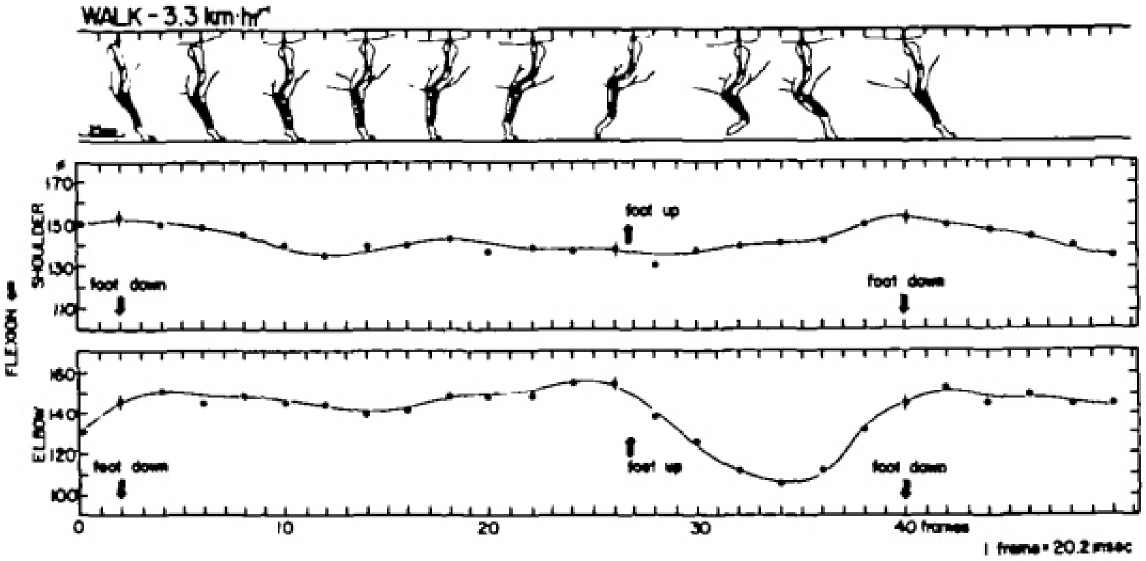


Figure 13 - Horse Walk at 3.3km/hr

Observing Figure 13 might be difficult to comprehend or make any sense of at first because there is a shoulder and elbow moving in motion. As can be seen, though, there is a defined time for when the foot leaves the ground and when the foot hits the ground. This defines the duty cycle of the gait. The duty cycles of various gaits are depicted in Figure 14. The black represents the percent of time the foot touches the ground, and the rest represents time in the air. If the duty cycle is over 50%, the gait is typically referred to as a walk. Otherwise, the gait is defined as a run.

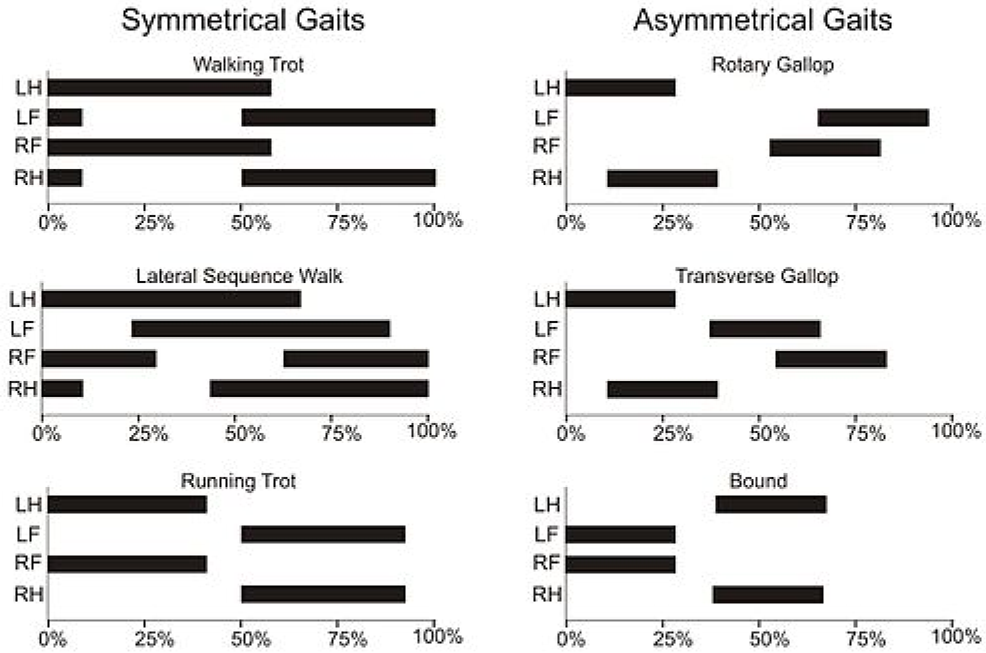


Figure 14 - Gaits represented through a duty cycle

2.1 Legged Robotic Systems

While our team approached the project from an innovative angle working to develop a number of innovative subsystems, the overall concept of a legged robot is nothing new. In fact, the community of legged robotics has well documented milestones in the development of legged machines summarized in Table 8.

Year	Inventor	Summary
1850	Chebyshev	Designs linkage used in early walking mechanism
1872	Muybridge	Uses stop-motion photography to document running animals
1893	Rygg	Patents human-powered mechanical horse
1945	Wallace	Patents hopping tank with reaction wheels that provide stability
1961	Space General	Eight-legged kinematic machine walks in outdoor terrain
1963	Cannon, Higdon and Schaefer	Control system balances single, double, and limber inverted pendulums
1968	Frank and McGhee	Simple digital logic controls walking of Phony Pony
1968	Mosher	GE quadruped truck climbs railroad ties under control of human driver
1969	Bucyrus-Erie Co	Big Muskie, a 15K ton walking dragline is used for strip mining
1977	McMahon and Greene	Human runners set new speed record on tuned track, compliance adjusted to mechanics of human leg.
1980	Hirose and Umetani	Quadruped machine climbs stairs and over obstacles using sensors
1980	Kato	Hydraulic biped walks with quasi-dynamic gait
1980	Matsuoka	Mechanism balances in the plane while hopping on one leg
1981	Miura and Shimoyama	Walking biped balances actively in three-dimensional space
1983	Sutherland	Hexapod carries human rider. Computer hydraulics and human share computing task.
1983	Odetics	Self-contained hexapod lifts and moves back end of pickup truck

Table 8- Milestones in the Development of Legged Robots (Adapted from Raibert, 1986)

These milestones were crucial to the development of modern day robotic systems. An outline of the more well-known and relevant robotic systems is provided in the following sections. These robots have served as inspiration in both the positive outcomes of the research and for further understanding the limitations in the modern day systems.

2.1.1 BigDog

BigDog developed by Boston Dynamics is a rough terrain robot measuring 3 feet long by 2.5 feet tall and weighs roughly 240 pounds. BigDog is arguably the world's most well known and most advanced quadruped. The project first began in 1986 in the MIT leg laboratory. The laboratory developed a series of robots in the 1980's and 1990's that moved and balanced dynamically. These robots contained 1,2 or 4 legs and were able to complete hopping, trotting and bounding gaits. They could also climb simple staircases, and jump over obstacles. Having demonstrated the capability of developing dynamically balanced legged systems the team then began work on BigDog, a self-contained quadruped robot. BigDog is a heavily funded project by the Defense Advanced Research

Projects Agency (DARPA). The goal of the project is to build an unmanned legged vehicle which can navigate rough terrain with mobility superior to existing wheeled or tracked vehicles. A picture displaying the system with the full payload can be seen in Figure 15.

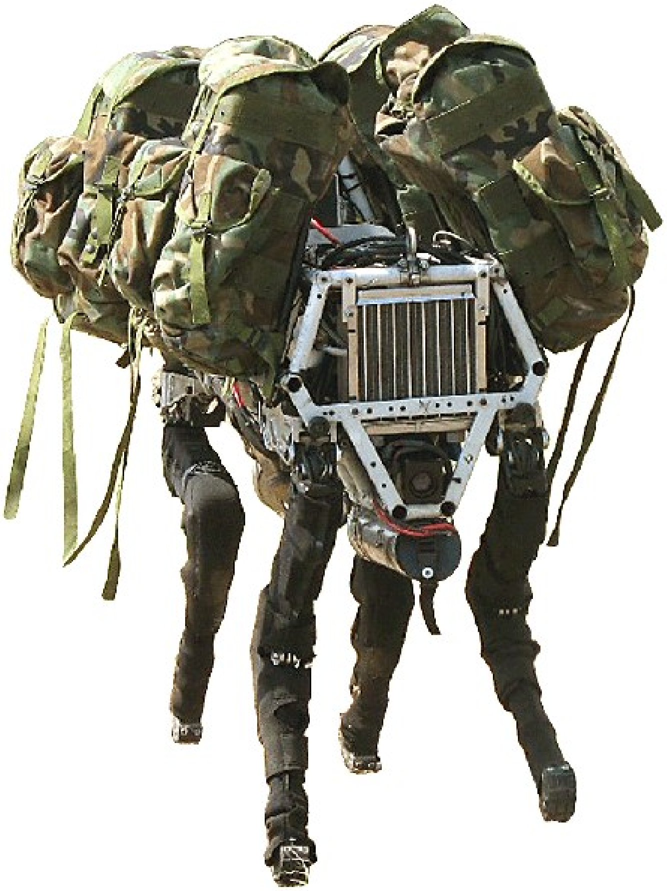


Figure 15 - BigDog System with Payload

BigDog has a self-contained processor which works to actuate and sense the controls and communications for the robot. The robot is able to provide power with the help of a two-stroke internal combustion engine which delivers approximately 15 horsepower. This engine operates a hydraulic pump which then actuates the joints of the robot's legs. The actuators are hydraulic cylinders which are operated by servo-valves. Each individual leg in the system has 4 active and one passive degree of freedom. The robot is able to crawl at speeds of 0.2 m/s, trot at speeds of 1.6 m/s, run with a trotting gait at speeds of 2 m/s, and briefly complete a bounding gait at 3.1 m/s.

BigDog is also able to traverse mud, snow and heavy inclines on a large variety of surfaces including rutted trials, rocky and loose scree. The robot is designed to carry payloads of up to 340lbs.

BigDog uses an array of sensors including an Inertial Measurement Unit (IMU) to measure the acceleration and altitude of the body. Additionally the robot also uses sensors at the joint to measure both the motion and force of the actuators. These sensors are crucial to providing real time data for the computer to process and calculate new gait algorithms (Raibert, 2008).

BigDog, while an impressive machine is not without a number of serious limitations. The first and most evident of these limitations is the noise which the system outputs during operation. During operation, BigDog's two-stroke gasoline engine outputs an undesirable loud noise, similar to the noise outputted by many lawnmowers. BigDog is also a robot which relies on a dedicated heavy control system. The system contains only one shock absorbing spring to act as a passive degree of freedom. This limits the mechanical compliance in the system and shows that the system does not mimic nature very well. Due this limited compliance, the actuators are expected to withstand heavy shock forces and limit the overall run time of the system. BigDog can only complete a running gait for a very short period of time due to this limited compliance. Finally, while BigDog can navigate a magnitude of complex terrain the robot is currently unable to walk up stairs. BigDog has clearly set a new precedent and achieved significant milestones in the field of legged locomotion, however, the necessity of having a robot to overcome all of these obstacles still exists.

2.1.2 LittleDog

BigDog has a younger brother by the name LittleDog, also developed by Boston Dynamics. LittleDog is also a quadrupedal self-contained robot but in a much smaller package. The robot measures 11.8 inches long, 7.1 inches wide and stands 5.5 inches tall. The robot weighs 4.9 pounds. This robot, unlike BigDog, has three motors per each leg. Unlike BigDog, this robot does not suffer from issues of underpowered motors. Instead, LittleDog has issues with motors which cannot spin fast enough to achieve all of the ideal gaits. LittleDog was designed as a research platform from which higher level gait algorithms and vision systems could be tested. In 2005, a LittleDog robot was entrusted to six robotics universities to improve on the controls algorithms and sensor systems originally developed in Phase I of this project. The teams have managed to develop higher level planning for the robot, higher accuracy dynamic models and stereovision integration (Kolter, et al., 2008). A picture of LittleDog walking over rough terrain can be seen in Figure 16.

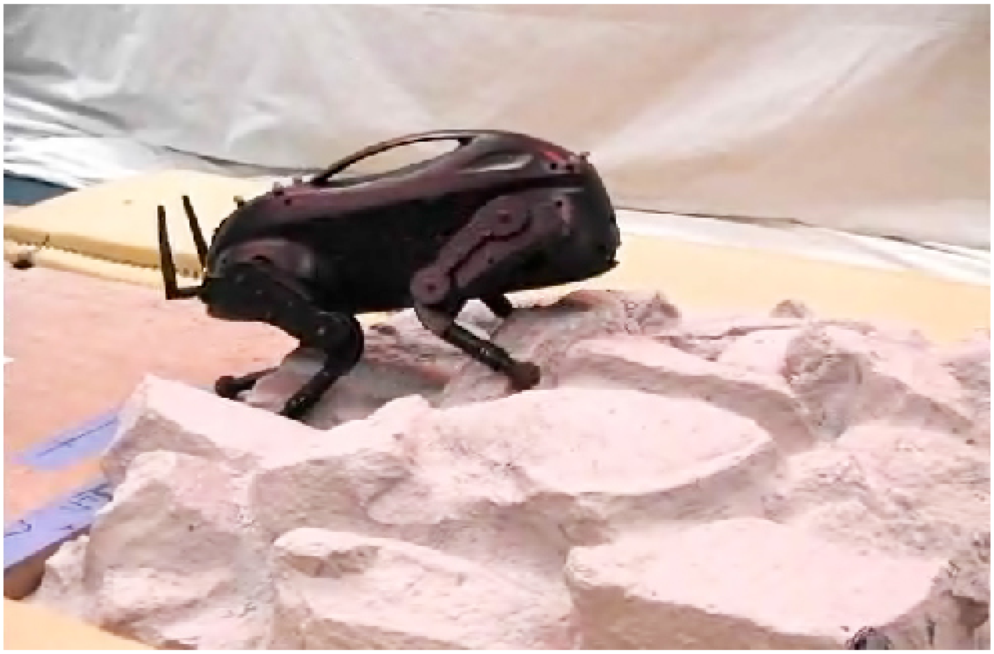


Figure 16 - LittleDog Robot Walking Over Rough Terrain

LittleDog, as well, is not without its limitations. The small form factor of LittleDog is both a positive and negative aspect. While the form factor allows for quicker and more agile maneuvers, it is also extremely limited when it comes to maneuvering over large obstacles. While the mechanics work perfectly in such a small form factor scaling up this system would call for un-proportionately larger motors and batteries. The LittleDog system is also extremely limited in the vision system on board (Kolter, et al., 2008).

2.1.3 MABEL

MABEL, shown in Figure 17, developed in conjunction with the University of Michigan and Carnegie Melon University, is a biped robot that utilizes the SLIP (Spring Loaded Inverted Pendulum) model, as explained in Chapter 2.5. The robot stands 1 meter tall at the hip and approximately 2 meters tall overall with a weight 60 kilograms. In addition to MABEL there is a sister robot called Thumper which is a unipod that has an exact copy of one of MABEL's legs.

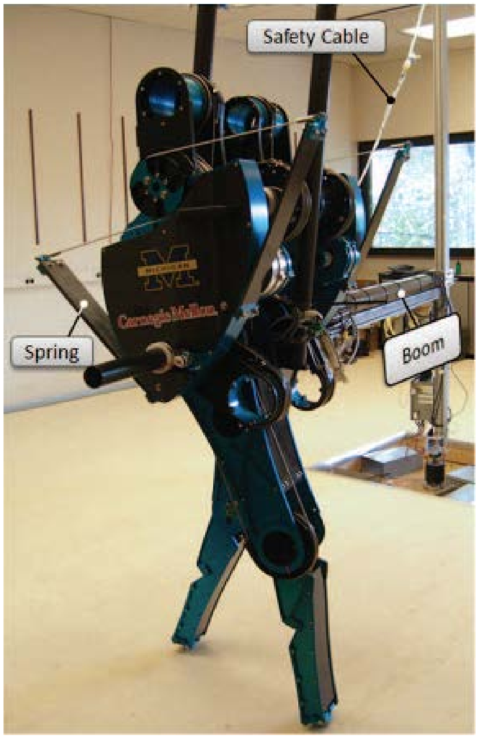


Figure 17 – MABEL

MABEL was developed to mimic the gait of a human, using the robots springs to act as tendons to recover power with each step. This was accomplished by the use of a number of unique mechanical designs and control techniques (Grizzle, 2008).

MABEL has two 2 degree of freedom legs, with each leg therefore having two motors to control the degrees of freedom. The unique aspect is that while the two degrees controlled are the knee and shoulder joints, each motor has no unique independent joint control. Rather, the system is reliant on both motors working together as a whole. The system is modeled as a single revolute joint and a prismatic joint going from the robots actual hip to its actual toe, as seen in Figure 18. Each motor moves both joints simultaneously, with one motor controlling the modeled link length and the other controlling the modeled link angle.

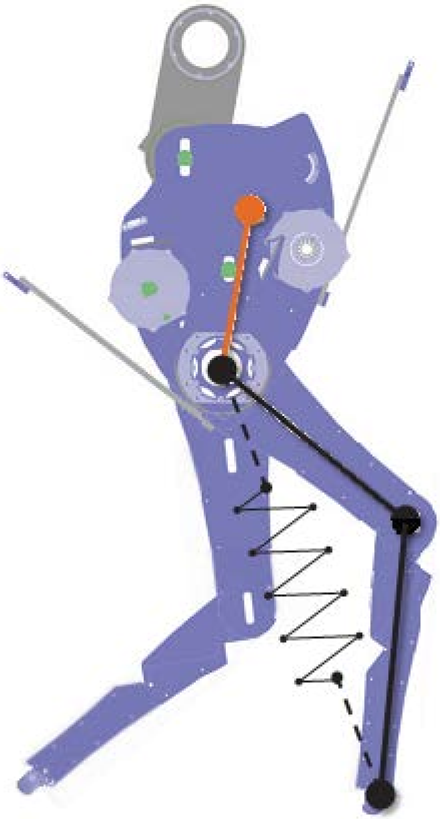


Figure 18 - Mabel Modeled as a Single Link with 2 DOF

The single imaginary link model and spring system are mechanically implemented using a series of 3 cable driven differentials. The first differential has the two inputs being the modeled link length motor and the spring themselves. This essentially creates the pogo stick that is inherent in the SLIP model. The output of this differential is then input into two other differentials that are also driven by the modeled angle motor. This allows both actual links of the legs to remain stationary relative to one another, preserving the imaginary link, but move relative to the actual hip.

MABEL, while able to recover energy and mimic human walking using the SLIP model, is not mobile. All the power and processing is done by a large cabinet of computers and power supplies that are just as large as the robot itself. The power and data are delivered to the robot through a tether. Additionally, MABEL has no way to stay upright without the lateral stabilization of the boom it's attached to. Hence, MABEL can only walk in circles (Grizzle, 2008).

2.2 Sensors

In order to safely navigate through an unknown environment, it is necessary for the robot to incorporate a vision system. This system is to serve two main purposes – to provide real-time visual data about the environment and locate obstacles in the environment. The first is accomplished with an on-board camera. The second is satisfied with a Light Detection and Ranging (LIDAR) sensor.

In addition to the vision system used to acquire data about the robot's environment, it is also necessary to incorporate sensors for feedback about the robot's position and orientation. These will allow the robot to accurately know the error of where the robot is versus where the robot should be. These sensors include an inertial measurement unit, quadrature encoders, potentiometers, and current sensors.

2.2.1 Camera

Camera systems are widely used in robotics for object recognition. In our case, a camera is needed to recognize a staircase and determine its location relative to the robot. Since standard staircases are of the same relative shape, a reasonable solution to detecting staircases is an algorithm that extracts key features of an object. There are two main algorithms commonly used to perform such a task – scale-invariant feature transform (SIFT) and speeded-up robust features (SURF).

SIFT, published in 1999, starts with extracting key points from reference images (Lowe, 2004). These key points provide a description of the desired object that can be used to identify an object in other images. The output from these reference images is stored in a database. To identify the object in a new image, the features of that new image are extracted and compared to the database via the Euclidean distance of the new features; the Euclidean distance is calculated via the Pythagorean formula. The algorithm then searches for small clusters of matches and, based on the number and location of the clusters and outliers found, the probability that the image contains the object is calculated. An example output of the SIFT algorithm is shown in Figure 19, where images of a phone and a shoe have been extracted.

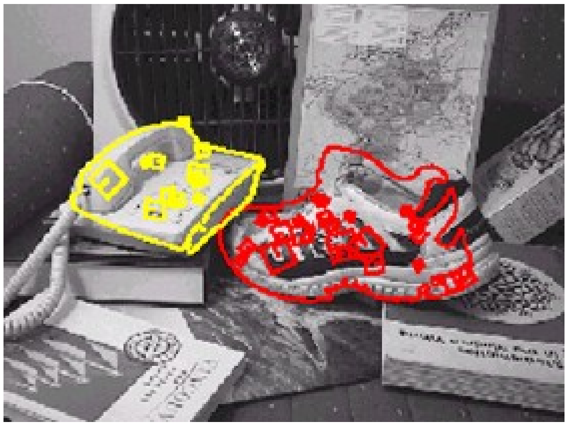


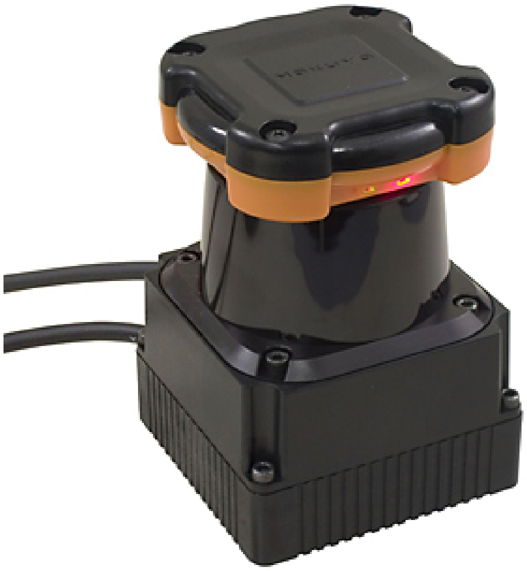
Figure 19 - Sample SIFT output

SURF, published in 2006, is largely based off of SIFT. However, it was designed to be faster and more robust than SIFT against different image transformations (such as rotations and skewing). SURF takes advantage of the Haar-like features algorithm, which was the feature extraction algorithm used in the first real-time face detectors. The algorithm essentially moves a rectangle across the image calculating the Haar-like features, which is the difference between the sums of the pixels inside that rectangle. This calculation indicates certain characteristics about the pixels in that rectangle that can be used to compare against other images. SIFT was further improved upon by implementing integral images, or summed area tables. These are essentially lookup tables used to quicken the process of computing the Haar-like calculations. The combination of these two sub-algorithms improves the feature extraction step of SIFT to near linear time.

Researching two of the most commonly used object recognition algorithms provided the team with the ability to decide upon which algorithm to implement for use on the robot.

2.2.2 LIDAR

A LIDAR is an optical sensing device that can be used to measure distances to objects within the range of the LIDAR. It works by sending pulses of light from an internal laser and measuring the time it takes for a specific pulse to return to the LIDAR. The distance to the object is calculated from that time and from the speed of light. There are typically three major components to a LIDAR. First, a laser is used to emit the aforementioned pulses outward. Second, a scanner is used to scan the laser to provide two dimensional distance data. Lastly, a photo detector is necessary to measure when a light pulse returns to the LIDAR. The LIDAR used in this project is the Hokuyo UTM-30LX.



Specifications	
Voltage	12.0V +/-10%
Current	0.7A (Rush current 1.0A)
Detection range	0.1m to approximately 60m (<30m guaranteed)
Laser wavelength	870nm, Class 1
Scan angle	270°
Scan time	25msec/scan (40.0Hz)
Angular resolution	0.25°
Interface	USB 2.0
Weight	8.2 oz (233 gm)

Figure 20 - Hokuyo UTM-30LX LIDAR and specifications

For this project, a LIDAR is needed to construct a three dimensional map of the environment. This map is to be used for path planning through a previously unknown environment using path planning algorithms.

2.2.3 IMU

An inertial measurement unit (IMU) is capable of measuring several things. Most IMUs include a three-axis accelerometer for measuring acceleration forces acting on the sensor and a gyroscope for measuring the sensors orientation with respect to the earth's axis (i.e. yaw, pitch, and roll). Applying this sensor to the robot allows for accurate measurement of the velocity at which the robot is moving and the orientation of the robot. The latter is important for balance control.

A major disadvantage to IMUs is that they are very prone to accumulated error. The errors accumulate due to the way IMUs continually add detected changes to previously calculated values. Therefore, the errors (regardless of size) are blindly added to the values every time step. This error can be reduced one of two ways: first, by including additional sensors to correct the error, such as a global positioning system; second, by implementing an error-reducing filter, such as a Kalman filter. Kalman filters serve to use measurements taken over time that include inaccuracies, and produce an output for the sensor that is closer to what the values should be.

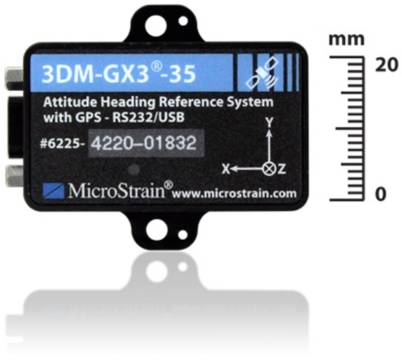

3DM-GX3®-35
Attitude Heading Reference System
with GPS - RS232/USB
#6225- 4220–01832
Y
X
Z
MicroStrain® www.microstrain.com
mm
20
0

Figure 21 - Microstrain 3DM-GX3 IMU

2.2.4 Quadrature Encoder

Rotary encoders are used to measure angular position, velocity, and acceleration. They are composed of a disc separated into black and white segments and a light detector that changes it output depending on whether it is pointed at a white or black segment. The angular position is measured in ticks of an encoder (i.e. the number of times the light detector measures a change in what it is pointed at).

Single disc encoders can only be used to measure displacement. Quadrature encoders, on the other hand, can measure displacement and direction of motion (clockwise or counter-clockwise). They work by using two discs offset by 90 degrees. The direction of rotation is determined by measuring the phase difference of the two discs (if the inner disc leads the outer disc, or if the outer disc leads the inner disc).

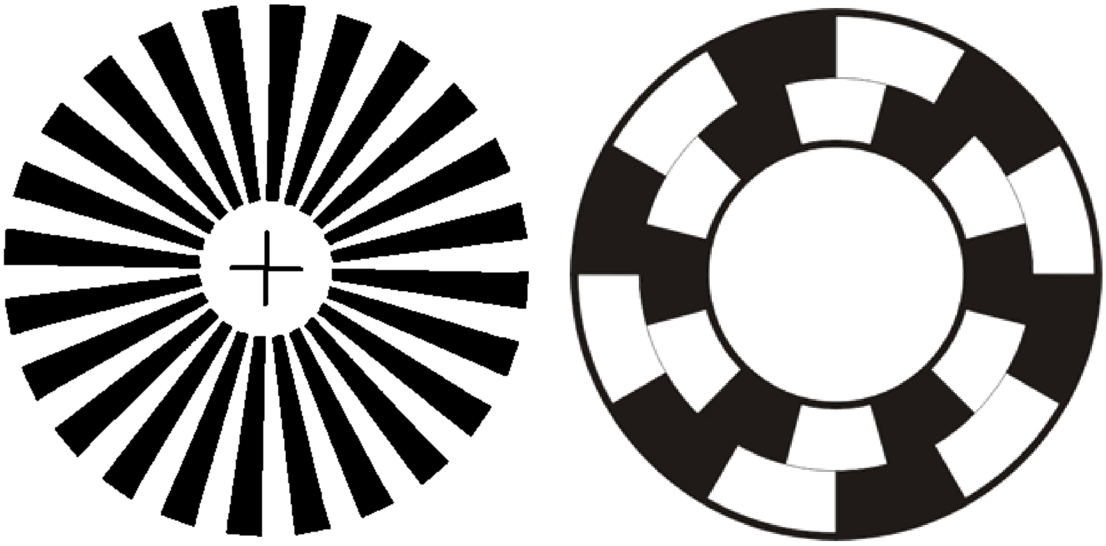


Figure 22 - Discs for a single encoder (left) and quadrature encoder (right)

2.2.5 Potentiometers

Potentiometers can be used to measure joint positions between two links or mechanical aspects of a robot. Potentiometers are variable resistance devices – the resistance of the potentiometer is determined by the rotational position of a shaft attached to the potentiometer. By mounting the base of a potentiometer to a robot and attaching the shaft to the joint of a link that moves rotationally, the change in resistance of the potentiometer can be used to calculate the change in angle of the link.

Figure 23 - Potentiometer

2.2.6 Current Sensors

Current sensors are devices used to measure electrical current (alternating and direct) and output a signal proportional to the strength of the current. For the robot, current sensors are required to determine when the robot's foot hits the ground. When this happens, a large current spike will occur in the leg due to the significant increase in weight applied against the motors driving the leg. This spike can be used to determine if the foot is on the ground when desired, and if the foot is on the ground when it shouldn't be. Current sensors are also used to ensure the robot does not overdraw on current while walking. This is to protect the sensors and controllers on the robot, as well as to make sure overly large quantities of heat are not given off. Figure 24 is an image of the current sensor used on the robot:

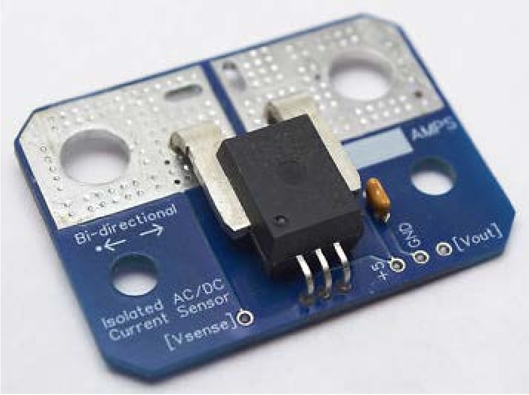


Figure 24 - 100 Amp AC/DC current sensor

2.3 Path Planning

There are several commonly used path planners in robotics, each with their advantages and disadvantages. The team researched A* (pronounced "A-star"), D*, and Dijkstra search algorithms. A* search is guaranteed to be complete and optimal, assuming a path exists. It works by keeping track of two lists of nodes. The first, called the open set, is a selective list of nodes the algorithm has yet to explore. The closed set is the set of nodes that have been completely explored. To be moved from the open set to the closed set, the node must have the lowest F-score of all nodes on the open list. The F-score is an estimation of the cost to travel from a node to the end node. It is based on the cost of travel to get to that node from the start node (which is completely known) and a heuristic used to estimate the distance from that node to the end node. This heuristic must be admissible (not overestimate the distance) in order for the path to be optimal. Once a node is on the closed set, the shortest path to that node has been found; therefore, the algorithm ends once the end node is on the closed set. The algorithm works as follows:

1. Add the starting square to the open set
2. Repeat the following until the end node is on the closed list, or the open list is empty (meaning a path does not exist):
   - a. Find the node on the open set with the lowest F-Score – this is the current node being looked at
   - b. Add that node to the closed set
   - c. For all nodes sharing an edge with that node, do the following:
     - i. If it is an obstacle, ignore it
     - ii. If it is not on the open list, add it to the open list and set its parent to be the current node, and calculate its F-Score
     - iii. If it is already on the open list, check to see if this path to that node is shorter than the previous path found
3. Trace the path backwards by starting at the end node and following the node parents until the start node is found. This is the shortest path.

D* is a variant of A*, standing for Dynamic A*. It is essentially a uniformed incremental search algorithm. It maintains the same open and closed sets as A*, but each node can be marked as raised (meaning it has a higher cost than the last time it was on the open list) or lowered (a lower cost than the last time it was on the open list). Also, unlike A*, D* starts with the end node and searches backwards, storing a back pointer from parent to child. This way, the exact cost to get to the end node is known exactly. When an obstacle is encountered, all points affected are placed back on the open list and marked as raised. The algorithm then checks if it can lower the nodes' cost. If not, the raised marker is propagated to every node that has a back pointer to the current node. This process continues, forming what's known as the "wave" of D*. When the cost can be reduced, a similar wave of the lowered marker is passed back along the back pointers, until the node that started the wave is reached. The algorithm continues with this process, expanding the node with the lowest cost, until the next node to expand is the start node.

Dijsktra's algorithm works similarly to A*, except it does not use a heuristic. The consequence of this is that the entire map is searched. The user is provided with more information than A* (for any node, you will know the shortest path from that node to all other nodes), but Dijkstra's algorithm is computationally more expensive.

2.4 Power Transmission

2.4.1 Cable Drive

A cable driven system is an efficient and effective way to transmit mechanical power. Rotary power is transmitted between two parallel shafts using the tension of a wire rope or cable acting on a pulley. Cable systems are an excellent choice for many mobile robots. Cable drive allows heavy components such as motors and gearboxes to be closer to the center of gravity of the robot, while allowing power to be transmitted out to the actuated systems. For its size, cables are stronger and lighter than chain. Another advantage over chain is that if properly tension cable systems have almost no backlash. This is because tension can be maintained on the slack side of the pulley without causing binding. Cable systems are also incredibly efficient, as there is no rubbing between teeth like in chain or gear systems (Norton, 2003).

2.4.2 Lead Screws

A lead screw is threaded rod that is intended to turn torque into a linear force. This is done by turning the screw and then attaching the desired linear components to a nut riding on the screw. Lead screws have several advantageous characteristics such as: large reductions in a small package, creating precise linear motion, and high resistance to backdrive. Because of the nature of the large sliding surface on the threads though, lead screws are inefficient and loose power to friction (Norton, 2003).

2.4.3 Gearboxes

In an application such as a robotic walker it is crucial to keep the motor at a high efficiency, high output torque and necessary RPM. As such the team explored a number of high efficiency gearbox options including planetary gearboxes, helical planetary gearboxes, cycloidal gearboxes and harmonic gearboxes.

PLANETARY AND HELICAL PLANETARY GEARBOXES

Planetary and helical planetary gearboxes, in most cases, are used to alter the motor input to a much lower output speed and higher output torque. The gear system generally consists of 3 unique types of gears: the sun gear, the planet gear, and the ring gear. This can be seen in Figure 25 where the red gear represents the sun gear, the green gears represent the planet gears and the blue gear represents the ring gear. Additionally, the 3 planet gears sit on a carrier (pictured yellow) which also rotates. The ring gear in this case is kept stationary as the rest of the system rotates. These gearboxes are able to offer a relatively large transfer of power in a small compact size. Additionally, the gearbox is perfectly in line which means that the profile of the motor is only extended in one direction when adding the gearbox (Norton, 2008).

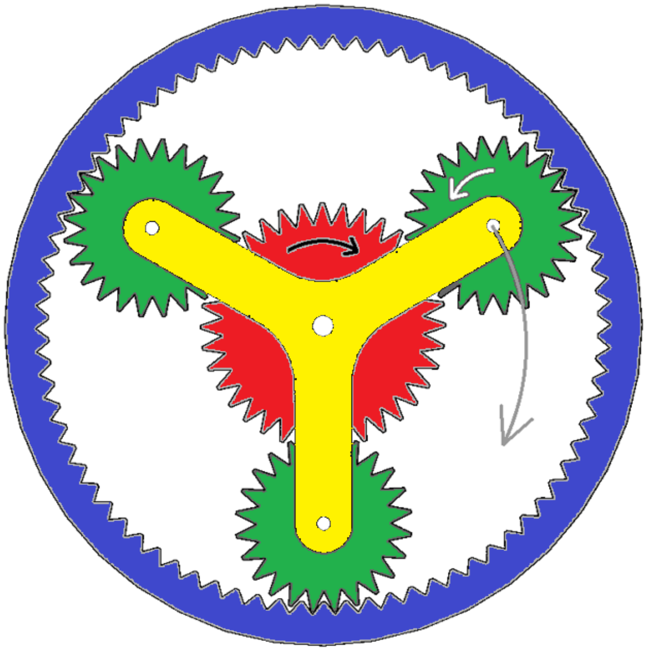


Figure 25 - Planetary Gearbox Transmission

Planetary and helical planetary gearboxes are both very similar with similar advantages, however share one distinct difference: the actual gears comprising the system. While planetary gearboxes have gears where the axes are all parallel, helical planetary gears do not. More specifically, they tend to follow a helical pattern about the gear. This allows for the gears to be able and handle more torque in the same size package. However, in many cases the gears will have more backlash as a result. A depiction of both a planetary and helical planetary gear can be seen in Figure 26 (Norton, 2008).

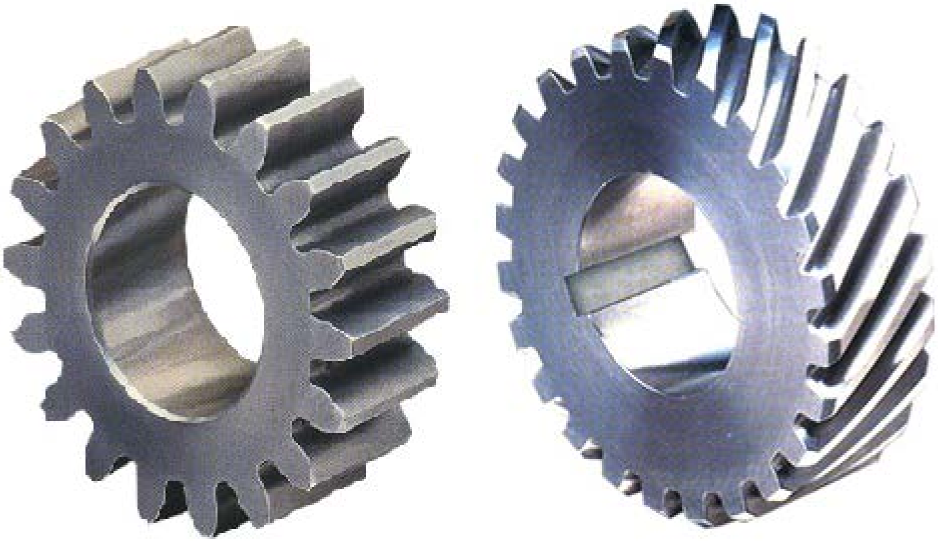


Figure 26 - Spur Gear (Left) and Helical Gear (Right)

CYCLOIDAL GEARBOXES

We also researched how cycloidal gearboxes work as well as their greatest advantages and disadvantages. As can be seen in Figure 27, a cycloidal gearbox is comprised of four major parts: the cam, the followers, the input shaft and the output shaft. The input shaft causes eccentric rotation of the cycloidal cam mechanism. The first track of the cam then engages with the followers. From here, the second track of the cam teeth engage with the cam followers and the output shaft allowing the eccentric rotation to turn into concentric rotation. Cycloidal gearboxes have a number of advantages compared to traditional planetary gearboxes. They have almost no backlash and that value stays constant throughout their entire life cycle. They are also able to be packaged in a flat pancake design which is much more compact than traditional planetary gearboxes. Finally, the design allows for extremely high shock load capacities. From a negative standpoint, however, this design offers much lower efficiency then well-built planetary gearboxes, has a much larger weight and in general is significantly more expensive.

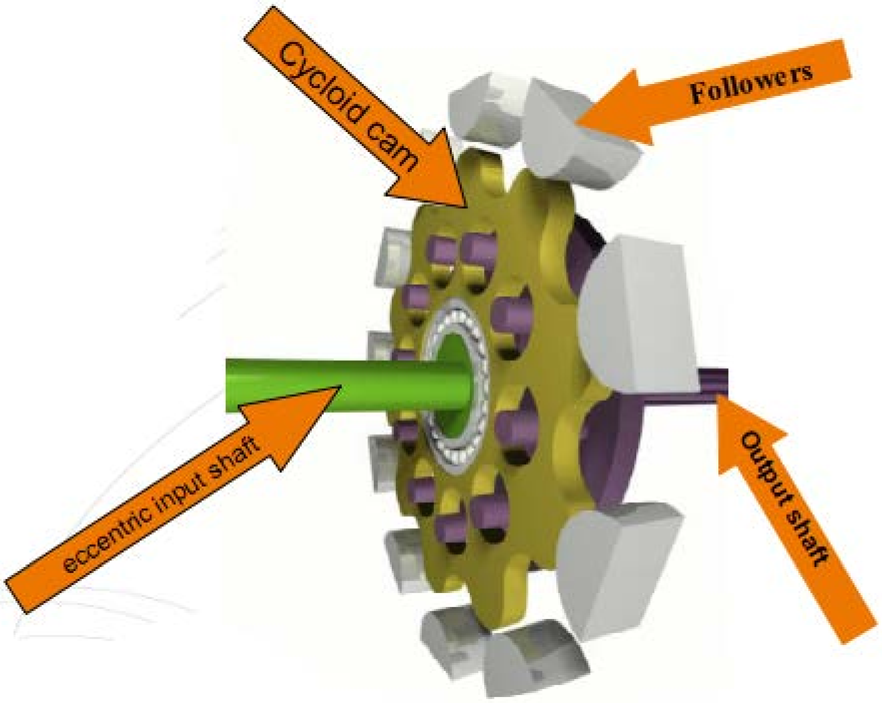


Figure 27 - Cycloidal gearbox major parts (Onvio, 2010)

HARMONIC GEARBOXES

Finally, we explored the advantages and disadvantages of harmonic gearboxes. Harmonic gearboxes are a special type of gear system relying on a fixed circular spline, a flex spline and a wave generator (Figure 28). The flex spline is attached to the output shaft and the wave generator is attached to the input shaft. The wave generator fits inside the flexible flex spline and causes eccentric rotation. Due to this eccentric rotation about the circular spline the output shaft is able to rotate with a predefined lower amount of rotation. The main advantage in harmonic gearboxes is their ability to have zero backlash, extremely high positional accuracy and low vibration in a very low profile design. Disadvantages of the system however include losing positional accuracy over time from high mechanical shock as well as a much increased cost relative to most other transmission systems (Tuttle, 1992).

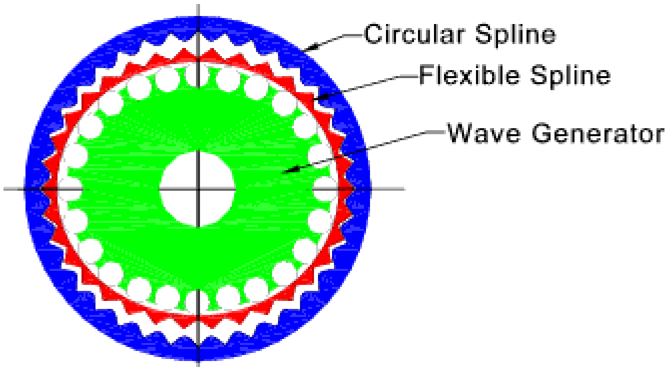


Figure 28 - Harmonic Gearbox Parts

2.5 SLIP Model

It is important to understand the mathematical model which can be used to represent the gait of any animal. Without understanding this fluid motion it would be useless to try and create gaits for any legged system. As can be seen in Figure 29, the SLIP model can be applied to many animals including humans. It is important to understand that the two links of each leg can be represented as a single link going through the step motion. Another unique feature of the SLIP model is that the center of gravity will stay relatively on the same plane throughout the entire step motion.

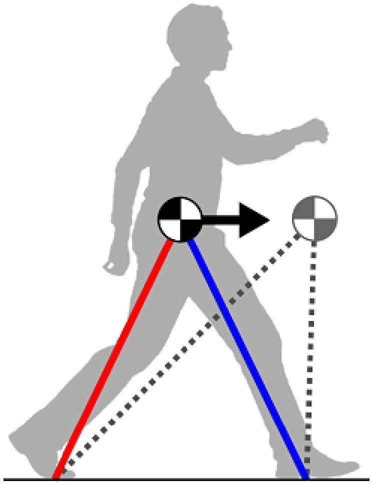


Figure 29 - SLIP in Biped

The SLIP model is the basis for some work with legged locomotion and needed to be carefully considered for its application in robotics. Simply put the SLIP model can be represented by a mass on a spring that acts like a pogo stick. The SLIP model provides a reasonable approximation to the center of mass motion of an animal in a running gait, regardless of type of legged animal, or the running gait employed (Full and Farley, 2000) (Blickhan and Full, 1993) (Dickinson et al., 2000) (Blickhan, 1989) (Farley and Ferris, 1998). Animals as diverse as humans and chickens have running gaits that can be approximated by the SLIP model (Blickhan et al., 1993) (Hutchinson et al., 2003). Successful running robots also exhibit SLIP model behavior, such as the Planar Hopper, ARL Monopod II and CMU Bowleg Hopper. (Raibert, 1986) (Ahmadi and Buehler, 1999) (Zeglin and Brown, 1998).

Throughout system design it was very important to realize how the springs react to the gaits of the robot. This spring compression and extension can be clearly seen in Figure 30. This change from power storage to power expansion is very important when thinking about both the power regeneration and mechanical compliance of the system.

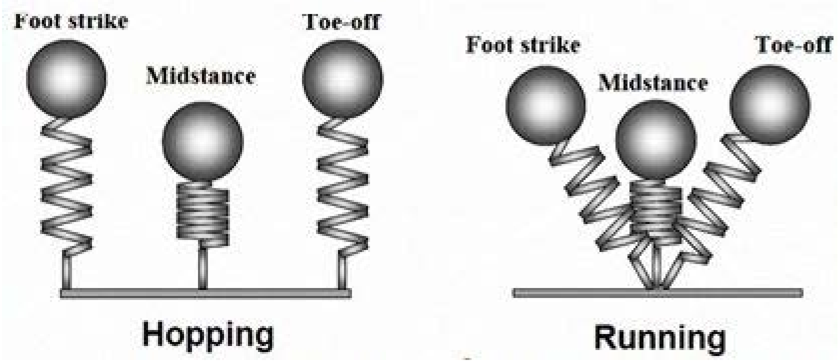


Figure 30 - Spring Mass System

The SLIP model also plays a very large role into the efficiencies of the systems that it is applied to. While this efficiency changes between animals, the ability to conserve lost energy is imperative in the animal's ability to go through different gaits. Applying the SLIP model to differing gaits can show exactly how important it is to keep in the design process. Reaching out to different gaits can truly explore how efficient the spring mass model can become. Rummel and Seyfarth proposed that a jointed leg could drastically reduce the speed required to gain this efficiency. Stating that a segmented leg could reduce the minimum speed required to produce stable running SLIP model from 3.5 m/s to a simple 1.5 m/s, exploring that any gain could be produced to recreate a version of SLIP (Rummel and Seyfarth, 2007). The ability to create equilibrium will greatly aid in the ability to produce a higher efficiency in the system allowing for less power to be required to keep the gait functioning properly.

With this background information the team moved forward with assessing task specification for the robot and designing its individual systems.

CHAPTER 3: METHODOLOGY

3.0 Task Specifications

In order to more fully and concretely define our problem statement we first came up with a set of design requirements for the system. These requirements were very focused and measurable to allow us to gauge the success of the project. The following were the defined goals of the project:

- Develop a robust mechanical base for legged locomotion research
- Robot must weigh less than 200 pounds.
- Must be capable of carrying a 30 pound payload the size of a cinder block.
- Dynamically imitate animal walking, trotting and bounding gaits
- Mechanically designed to be physically capable of walking, trotting and bounding gaits
- Develop compliant spring system allowing the robot to reproduce a SLIP model
- Robot must be untethered
- Recover and monitor energy lost with each step
- Achieve a statically stable 5ft/sec walking gait
- Develop a mechanically complaint ankle system able to recreate the "toe-to-heel" movement of quadrupedal animals
- Each leg must contain three degrees of freedom called the "hip", "shoulder", and "knee"
  - The "hip" must provide the leg with camber capabilities for dynamic stability and turning
  - The "shoulder" must be the top link of the 2 link system
  - The "knee" must be the bottom link of the 2 link system.
- Develop an flexible midsection of the robot that has 2 degrees of freedom.

- Develop software to learn the robot's surroundings using:
    - IMU for robot stability
    - LIDAR for obstacle avoidance
    - Camera for staircase recognition
- Demonstrate employment of A* search algorithm for path planning on the robot side
- Must contain a simple network communication protocol to allow for communication between all on- and off-board processors.

3.0 Full Mechanical Design

In order to develop a complete mechanical system the team used SolidWorks for all of their necessary drafting needs. The team was able to work through multiple problems by carefully modeling every component in the system. SolidWorks proved to be an excellent modeling software from which we were able to develop full mechanical drawings. Upon completion of the full mechanical design, the team separated the robotic system into different significant mechanical subassemblies. Great care was taken to design for all of the task specifications outlined in Section 3.0. A full view of the completed CAD model with identified subsystems can be seen in Figure 31. The identified systems will be explained in further depth through the rest of Chapter 3.

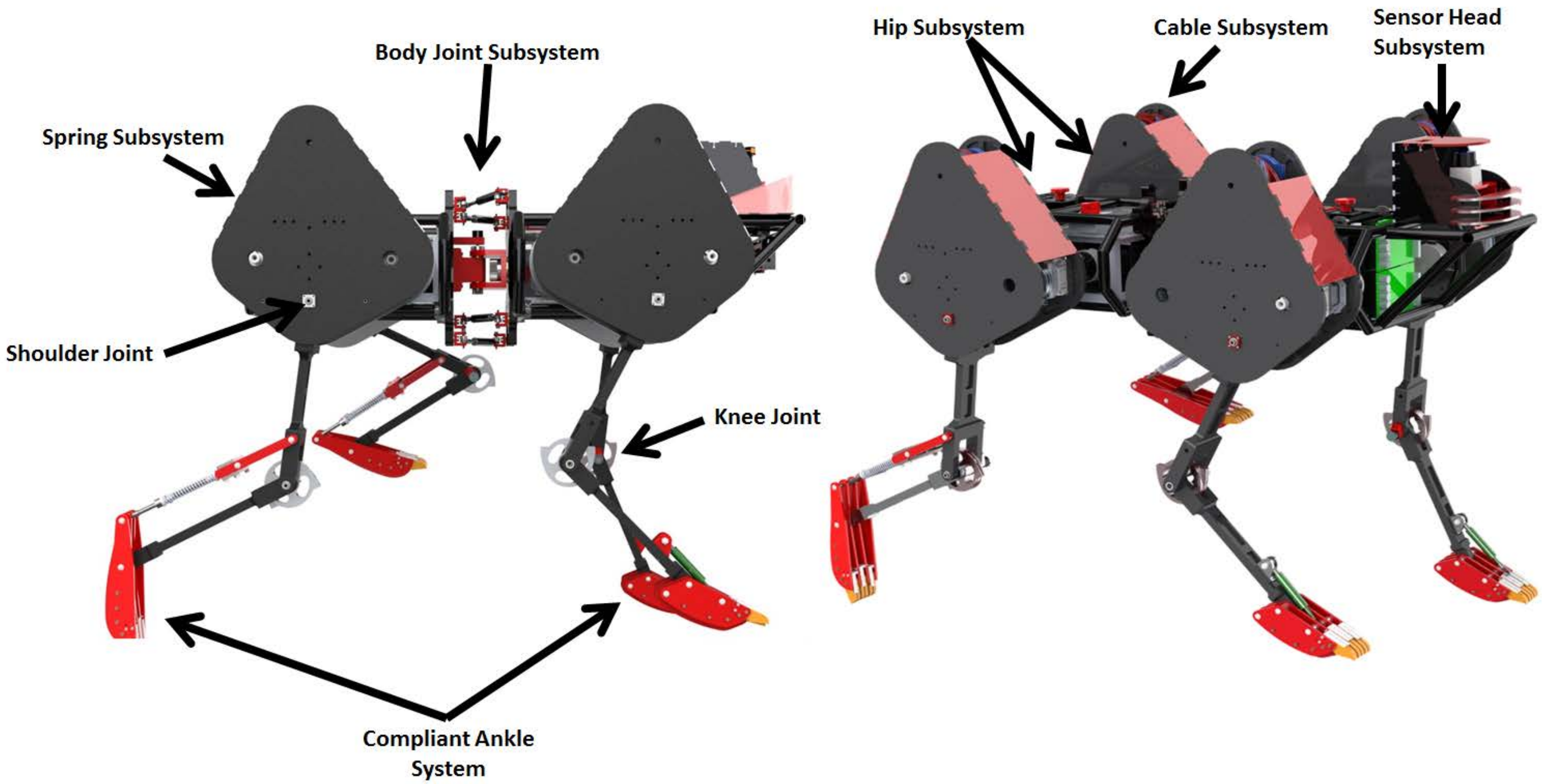


Figure 31 - Finalized CAD Model with Identified Subsystems

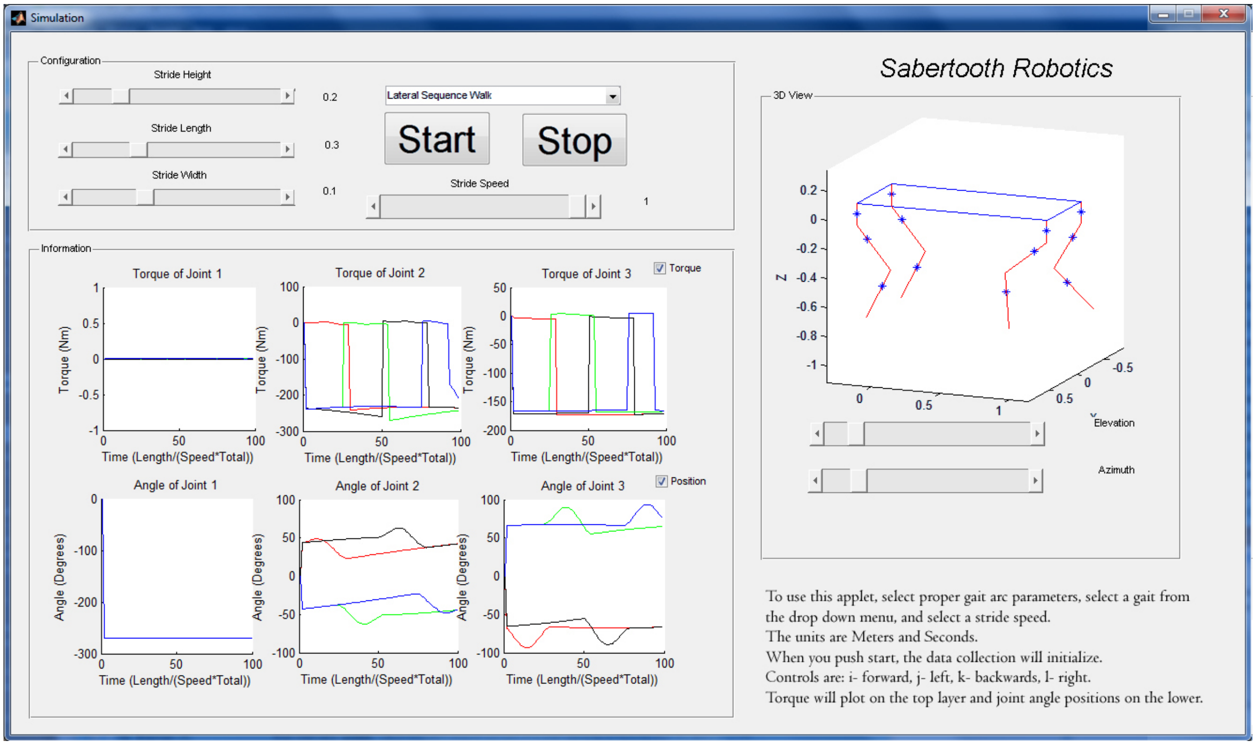
3.1 Modeling and Simulation

3.1.1 MATLAB Simulation

MATLAB is a numerical computing environment that gives powerful matrix manipulation, data plotting, algorithm implementation, and graphical interface capabilities. The software is intended to be used for numerical computation because it visualizes and declares everything as a matrix. The MATLAB language was developed for quick prototyping through a modified syntax of FORTRAN F77. It is widely used due to its abilities to manipulate large quantities of data and visualize them; being used in industries from economics to engineering as well as research in academic institutions.

MATLAB was chosen to create a visualized simulation of Sabertooth because of its effective ability to manipulate large amounts of data. This data was used to calculate motor requirements necessary to have Sabertooth walk at our desired speed of 5 ft/s as well as gait generation that would shows the ranges of motion that were required from the legs.

Figure 32 shows the fully functional visual simulation of the Sabertooth robotics platform. The user controls the robot through the I, J, K, and L keys (in configuration to arrows). The user may also select from various parameters such as the type of gait and the max distance the gaits arc achieves. As the robot moves, the simulation can output graphs with joint angle positions, velocities, and accelerations as well as Cartesian end effectors positions, velocities, and accelerations, and lastly torques applied to each joint.



Stride Height 0.2

Stride Length 0.3

Stride Width 0.1

Lateral Sequence Walk

Stride Speed 1

☑ Torque

☑ Position

To use this applet, select proper gait arc parameters, select a gait from the drop down menu, and select a stride speed.
The units are Meters and Seconds.
When you push start, the data collection will initialize.
Controls are: i- forward, j- left, k- backwards, l- right.
Torque will plot on the top layer and joint angle positions on the lower.

Figure 32 - The MATLAB Simulation

This simulation proved crucial to more fully understand the system and understand the required motion for the mechanical design. Simulation is essential in a robotic platform such as this. Actively calculating how the robot should be moving and being able to graphically visualize it is a quintessential first step when attempting to develop a system which relies heavily on accurate calculations.

3.1.2 Denavit-Hartenberg Parameters

The Denavit-Hartenberg (DH) parameters are a commonly used coordinate transformation system that allows selecting specific frames of reference for a rigid body. These are especially important when trying to realize and simulate a body translating and rotating in 3D space. DH parameters are a systematic approach to coordinate frame representations instead of a more arbitrary approach. In each homogeneous transformation matrix, there are 4 transformations made:

d – Offset along previous z to the common normal (translation)

θ – Angle about previous z, from old x to new x (rotation)

a – Length of the common normal (translation)

α – Angle about common normal, from old z axis to new z axis (rotation)

DH parameters are not the only convention for coordinate systems; however they are the most standardized approach in robotic systems. This is due mostly to their unambiguity in representing positions of links and joints. By drawing a full diagram of the coordinate frame of the robot(Figure 33) and understanding that the Θ parameter will be changing with time, t, the team was able to conclude with Table 9, the DH parameter representation of each joint on a leg.

Joint	**A**	**α**	**D**	**Θ**
1	.15m	90º	.064m	θ_1(t)
2	.381m	0º	0m	θ_2(t)
3	.381m	0º	0m	θ_3(t)

Table 9 - DH Parameters of Sabertooth leg

Expressing these in a mathematical notation through the use of a 4x4 matrix, Table 10 represents Link 1 or ${}_0^1T$, Table 11represents Link 2 or ${}_1^2T$, and Table 12 represents Link 3 or ${}_2^3T$.

$$\begin{bmatrix} \cos(\theta_1(t)) & 0 & \sin(\theta_1(t)) & 0.15\cos(\theta_1(t)) \\ \sin(\theta_1(t)) & 0 & -\cos(\theta_1(t)) & 0.15\sin(\theta_1(t)) \\ 0 & 1 & 0 & 0.064 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Table 10 - Link 1 Transformation Matrix

$$\begin{bmatrix} \cos\left(\theta_2(t)\right) & -\sin\left(\theta_2(t)\right) & 0 & 0.381\cos\left(\theta_2(t)\right) \\ \sin\left(\theta_2(t)\right) & \cos\left(\theta_2(t)\right) & 0 & 0.381\sin\left(\theta_2(t)\right) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Table 11 - Link 2 Transformation Matrix

$$\begin{bmatrix} \cos\left(\theta_3(t)\right) & -\sin\left(\theta_3(t)\right) & 0 & 0.381\cos\left(\theta_3(t)\right) \\ \sin\left(\theta_3(t)\right) & \cos\left(\theta_3(t)\right) & 0 & 0.381\sin\left(\theta_3(t)\right) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Table 12 - Link 3 Transformation Matrix

Figure 33, depicts a representation of the coordinate transformation for a leg on Sabertooth. Arrows show the direction around the Z axis which a positive θ value would rotate the coordinate system. Each of Sabertooth's leg joints are expressed as revolute joints.

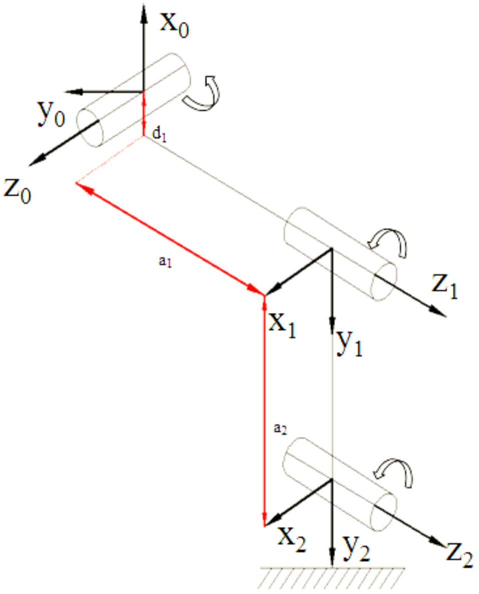


Figure 33 - DH Parameters of Sabertooth

The DH parameter model shown in Figure 33 was used for all four legs on Sabertooth. DH parameters were also used to understand the relative position of the IMU, LIDAR, and camera for understanding the perspective, position, and orientation is relevant with reference to the body for more accurate modeling. The IMU's placement was especially important to be placed as closely as possible to the actual revolute joint of the robot to properly analyze accelerations seen at the body joint. These parameters were then further used to begin modeling the forward and inverse kinematics of the system.

3.1.3 Forward Kinematics

Forward Kinematics is a computation of Cartesian position and orientation through rigid bodies as a function of joint angles. The robot was broken down into a body, hips, shoulders, and elbows, as shown in Figure 34. Forward kinematics allowed the team to plot the system in MATLAB.

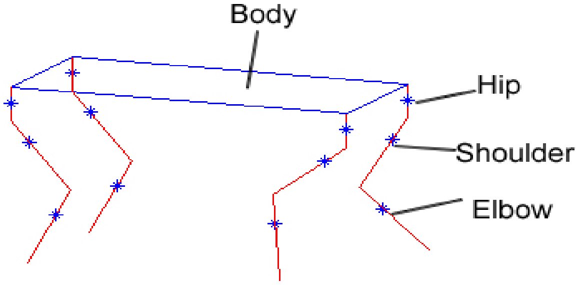


Figure 34 - MATLAB model of Sabertooth

Through matrix multiplication, the X, Y and Z coordinate of any given link's end effector could be found.

$$
{}^{n-1}T_n = \left[\begin{array}{ccc|c} \cos\theta_n & -\sin\theta_n\cos\alpha_n & \sin\theta_n\sin\alpha_n & r_n\cos\theta_n \\ \sin\theta_n & \cos\theta_n\cos\alpha_n & -\cos\theta_n\sin\alpha_n & r_n\sin\theta_n \\ 0 & \sin\alpha_n & \cos\alpha_n & d_n \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{ccc|c} & R & & T \\ \hline 0 & 0 & 0 & 1 \end{array}\right]
$$

Figure 35 - Coordinate Transformation

The following shows the mathematical computations performed to solve the kinematics of a given leg's link number. To save computational time, a recursive function designed to set the current link number's coordinates to a memory address for a given link is used. The full mathematical analysis for Sabertooth can be found in Appendix A: Maple Code of Forward Kinematics.

$$ {}_{0}^{1}T = \begin{bmatrix} \cos(\theta_1(t)) & 0 & \sin(\theta_1(t)) & 0.15\cos(\theta_1(t)) \\ \sin(\theta_1(t)) & 0 & -\cos(\theta_1(t)) & 0.15\sin(\theta_1(t)) \\ 0 & 1 & 0 & 0.064 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}_{0}^{2}T = {}_{0}^{1}T\,{}_{1}^{2}T = $$

$$ \begin{bmatrix} \frac{1}{2}\cos(\theta_1(t)-\theta_2(t)) + \frac{1}{2}\cos(\theta_1(t)+\theta_2(t)) & -\frac{1}{2}\sin(\theta_1(t)+\theta_2(t)) + \frac{1}{2}\sin(\theta_1(t)-\theta_2(t)) & \sin(\theta_1(t)) & 0.1905\cos(\theta_1(t)-\theta_2(t)) + 0.1905\cos(\theta_1(t)+\theta_2(t)) + 0.1500\cos(\theta_1(t)) \\ \frac{1}{2}\sin(\theta_1(t)+\theta_2(t)) + \frac{1}{2}\sin(\theta_1(t)-\theta_2(t)) & -\frac{1}{2}\cos(\theta_1(t)-\theta_2(t)) + \frac{1}{2}\cos(\theta_1(t)+\theta_2(t)) & -\cos(\theta_1(t)) & 0.1905\sin(\theta_1(t)+\theta_2(t)) + 0.1905\sin(\theta_1(t)-\theta_2(t)) + 0.1500\sin(\theta_1(t)) \\ \sin(\theta_2(t)) & \cos(\theta_2(t)) & 0. & 0.06400 + 0.3810\sin(\theta_2(t)) \\ 0 & 0 & 0 & 1. \end{bmatrix} $$

$$ {}_{0}^{3}T = {}_{0}^{1}T\,{}_{1}^{2}T\,{}_{2}^{3}T = $$

$$ \Big[\Big[\tfrac{1}{2}\cos(\theta_1(t)-\theta_2(t)-\theta_3(t)) + \tfrac{1}{2}\cos(\theta_1(t)+\theta_2(t)+\theta_3(t)), -\tfrac{1}{2}\sin(\theta_1(t)+\theta_2(t)+\theta_3(t)) + \tfrac{1}{2}\sin(\theta_1(t)-\theta_2(t)-\theta_3(t)), \sin(\theta_1(t)), 0.1500\cos(\theta_1(t)) + 0.1905\cos(\theta_1(t)-\theta_2(t)-\theta_3(t)) + 0.1905\cos(\theta_1(t)+\theta_2(t)+\theta_3(t)) + 0.1905\cos(\theta_1(t)-\theta_2(t)) + 0.1905\cos(\theta_1(t)+\theta_2(t))\Big], $$
$$ \Big[\tfrac{1}{2}\sin(\theta_1(t)+\theta_2(t)+\theta_3(t)) + \tfrac{1}{2}\sin(\theta_1(t)-\theta_2(t)-\theta_3(t)), -\tfrac{1}{2}\cos(\theta_1(t)-\theta_2(t)-\theta_3(t)) + \tfrac{1}{2}\cos(\theta_1(t)+\theta_2(t)+\theta_3(t)), -\cos(\theta_1(t)), 0.1500\sin(\theta_1(t)) + 0.1905\sin(\theta_1(t)+\theta_2(t)+\theta_3(t)) + 0.1905\sin(\theta_1(t)-\theta_2(t)-\theta_3(t)) + 0.1905\sin(\theta_1(t)+\theta_2(t)) + 0.1905\sin(\theta_1(t)-\theta_2(t))\Big], $$
$$ \Big[\sin(\theta_2(t)+\theta_3(t)), \cos(\theta_2(t)+\theta_3(t)), 0., 0.3810\sin(\theta_2(t)+\theta_3(t)) + 0.06400 + 0.3810\sin(\theta_2(t))\Big], $$
$$ \Big[0., 0., 0., 1.\Big]\Big] $$

3.1.4 Inverse Kinematics

Inverse kinematics performs the opposite function of forward kinematics. Inverse kinematics allowed the team to input a position and output a set of joint angles. With this information we could control input end effector positions and control the robot to the indicated positions with individual joint control. The inverse kinematics was solved using a geometrical approach. The equations used to solve for the angle positions with respect to the desired end effector position can be seen below.

$$ \theta_1 := \operatorname{atan2}\left(\mathrm{Desired}_z, \mathrm{Desired}_x\right) + \frac{\pi}{2} $$

$$ \theta_2 := \operatorname{atan2}\left(\mathrm{Desired}_y, \sqrt{\mathrm{Desired}_x^{\,2} + \mathrm{Desired}_z^{\,2}} - a_1\right) - \operatorname{atan2}\left(a_3\cdot\sin(\theta_3), a_2 + a_3\cdot\cos(\theta_3)\right) $$

$$ \theta_3 := \operatorname{atan2}\left[\pm\cdot\left[\frac{\sqrt{1 - \left[\left[\left(\sqrt{\mathrm{Desired}_x^{\,2} + \mathrm{Desired}_z^{\,2}} - a_1^{\,2}\right)^2 + \mathrm{Desired}_y^{\,2}\right] - a_2^{\,2} - a_3^{\,3}\right]}}{2\cdot a_2\cdot a_3}\right]^2, \frac{\left(\sqrt{\mathrm{Desired}_x^{\,2} + \mathrm{Desired}_z^{\,2}} - a_1^{\,2}\right)^2 + \mathrm{Desired}_y^{\,2} - a_2^{\,2} - a_3^{\,3}}{2\cdot a_2\cdot a_3}\right] $$

Knowing the desired Cartesian coordinates that we wanted any leg to be at, the three equations derived above allowed us to perform the inverse kinematics. The inverse kinematics were necessary in calculating the range of motion necessary for our legs to achieve a particular gait at a desired speed. The output of the joint angles going through a standard walking gait can then be actively plotted in MATLAB (Figure 36).

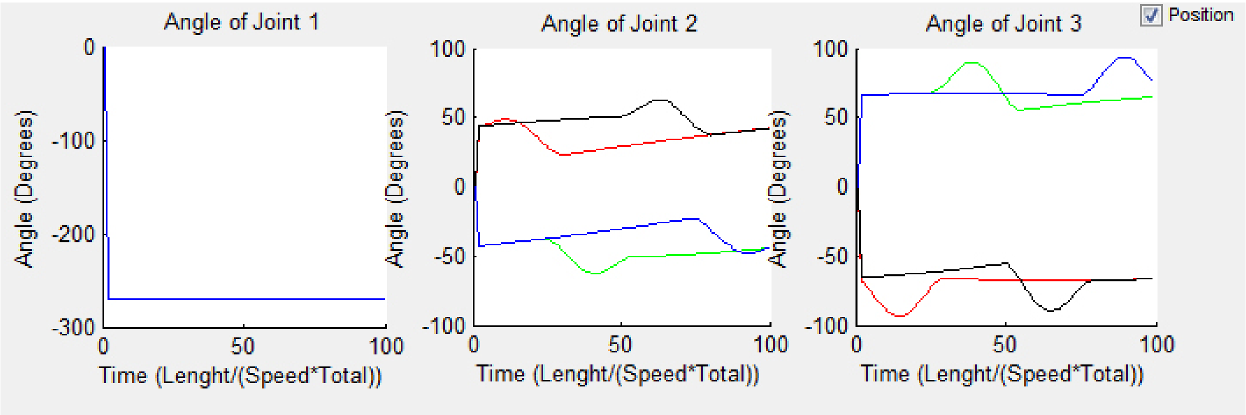
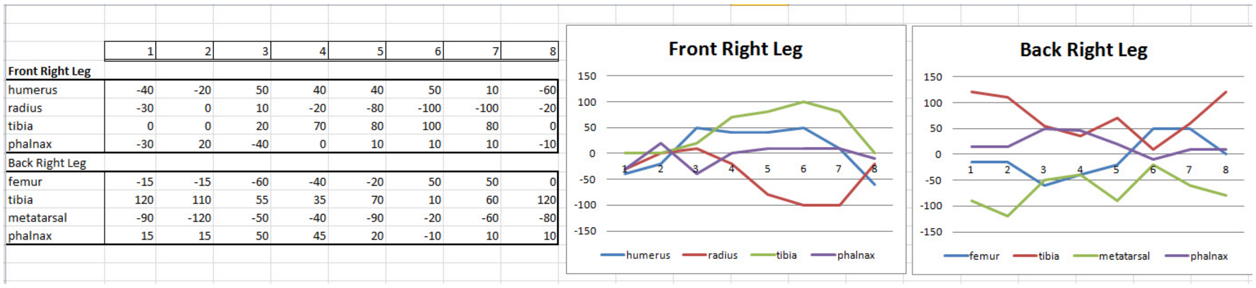


Figure 36 - Joint Angles through a walking gait

For any particular parameters for an arc (height and length when walking straight forward), analysis can also be conducted to solve stride length (length=0.32m and height=0.21). This arc gives us a large range of motion over obstacles and allows us to move 1 meter through three gaits.

3.1.5 Gait Generation

Taking certain coordinates and simulating them in MATLAB was the first step in figuring out how to better model these unique motions. The team first carefully watched stop motion frame photography of a greyhound walking through a standard gait. Using these coordinate points, the team created an EXCEL graph to simulate the robot going through a gait (Figure 37).



	1	2	3	4	5	6	7	8
Front Right Leg								
humerus	-40	-20	50	40	40	50	10	-60
radius	-30	0	10	-20	-80	-100	-100	-20
tibia	0	0	20	70	80	100	80	0
phalnax	-30	20	-40	0	10	10	10	-10
Back Right Leg								
femur	-15	-15	-60	-40	-20	50	50	0
tibia	120	110	55	35	70	10	60	120
metatarsal	-90	-120	-50	-40	-90	-20	-60	-80
phalnax	15	15	50	45	20	-10	10	10

Figure 37 - Modeling in excel and expressing as functions

Using Excel's equation automatic trend-line solver, solving for a sixth order polynomial function that represented the graphs with an R^2 value above .8 was achieved. This was then used with a MATLAB simulation model that allowed a visual representation of a quadruped going through a gait.

Though for visuals it proved that simulating a gait was plausible, the team quickly learned that the precision and physics was not able to be calculated from these polynomial functions such as precise foot placement. Instead, gaits were then further refined to mimic typical quadrupedal motion.

Starting by creating a gait loop which represents going through a gait from 0% to (final cycle+duty cycle)%. The duty cycle is represented by the max percentage of time the foot is spent off the ground. The final cycle is the percentage at which the last foot in the series leaves the ground. For example, in the bound gait the duty cycle is approximately 35% and the final cycle is approximately 70%- hence the loop will be from 0% to 105%. Lastly creating an offset for each leg is set as the first contact with the ground for the foot.

Looking at the bound gait- LF and RF offset's will be 0% and RH and LH will be 38%. As the loop is iterated, there will be a time when the leg is off the ground, less than the offset and greater than the offset+duty cycle, 0% to 38% and 70% to 105%, where the leg will be in the air. 38% to 70% it will be on the ground. To calculate the desired x, y, z of the foot going through the gait the following formulas for the x y and z displacement were developed and implemented. Explanation and variable definitions are depicted in Figure 38.

$$x(i) := \begin{bmatrix} i\cdot\left(\dfrac{F_x(X_0 - X_1)\cdot \text{width}}{\text{duty}}\right), \text{if}(\text{offset} < i) \wedge (i < \text{offset} + \text{duty}) \\ -i\cdot\left(\dfrac{F_x(X_0 - X_1)\cdot \text{width}}{\text{duty}}\right), \text{if}(\text{offset} > i) \vee (i > \text{offset} + \text{duty}) \end{bmatrix}$$

$$y(i) := \begin{bmatrix} i\cdot\left(\dfrac{F_y(Y_0 - Y_1)\cdot \text{length}}{\text{duty}}\right), \text{if}(\text{offset} < i) \wedge (i < \text{offset} + \text{duty}) \\ -i\cdot\left(\dfrac{F_y(Y_0 - Y_1)\cdot \text{length}}{\text{duty}}\right), \text{if}(\text{offset} > i) \vee (i > \text{offset} + \text{duty}) \end{bmatrix}$$

$$z(i) := \begin{bmatrix} \dfrac{\sin[(i - \text{offset})\cdot 2\pi]}{\text{duty}}\cdot 2\cdot \dfrac{\text{height}}{\text{duty}}, \text{if}(\text{offset} < i) \wedge (i < \text{offset} + \text{duty}) \\ 0, \text{if}(\text{offset} > i) \vee (i > \text{offset} + \text{duty}) \end{bmatrix}$$

Figure 38 - Equations to express the arc within a gait for a desired x, y, and z coordinate.

Duty: Time the foot is allotted contact with the ground.

Offset: The time instance at which the foot first comes into contact with the ground preceding time in the air.

Width, Height, Length: The preset parameters of the arc.

Total : Duty added to the last *Offset* in the gait

i : The value in the domain from 0 to *Total* that represents the location of the gait.

$F_x(X_0,X_1),F_y(Y_0,Y_1)$: Functions that define unit vectors in the directions of input coordinates

These equations allowed for the full control and simulation of Sabertooth going through a gait, allowing for examination of speed and various arc parameters. This helped us understand ideal conditions as we walked through a gait and check to make sure as the robot performed a certain gait all angles were plausible and could be achieved.

3.1.6 Newton-Euler Recursive Function

The Newton-Euler function (Figure 39) combines the rotational and translational dynamics of a rigid body. Through simulation we treated our body as a rigid body because of the way our control system works and the sheer complexity of our internal spring system. Figure 40 identifies the MATLAB simulation calculating torques applied to the joints. The torques calculated were then used in motor calculations. Figure 39 depicts a generic gait pattern used

$$\begin{pmatrix}\mathbf{f}\\ \boldsymbol{\tau}\end{pmatrix} = \begin{pmatrix} m\mathbf{I} & 0 \\ 0 & \mathbf{J}_c\end{pmatrix}\begin{pmatrix}\ddot{\mathbf{q}}\\ \dot{\boldsymbol{\omega}}\end{pmatrix} + \begin{pmatrix}0\\ \boldsymbol{\omega}\times\mathbf{J}_c\boldsymbol{\omega}\end{pmatrix},$$

Figure 39 - Newton-Euler equations

f : total force acting on the center of mass

m : mass of the body

I : the identity matrix

q : acceleration of the center of mass

τ : total torque (or moment) acting about the center of mass

J_C : moment of inertia about the center of mass

w : angular velocity of the body

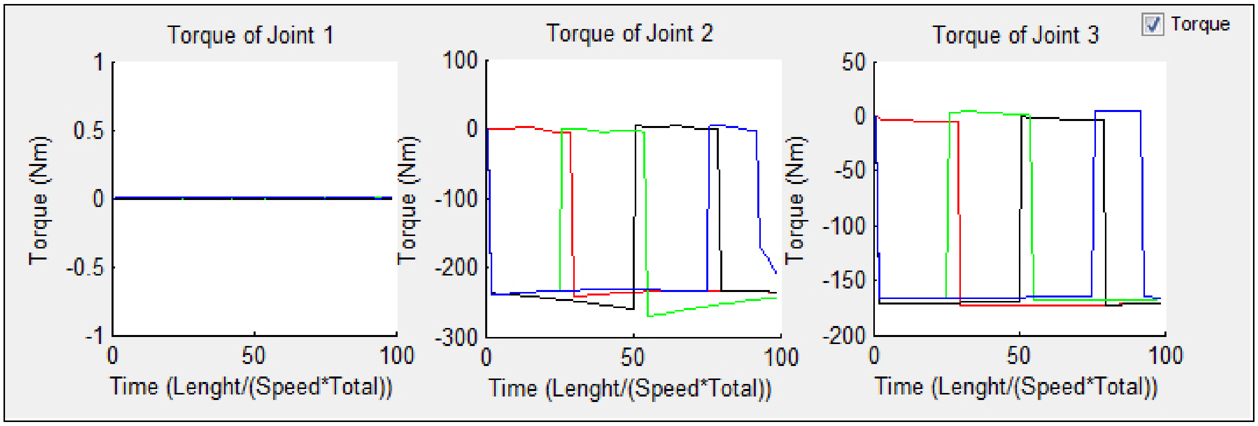


Figure 40 - Torques at specific joints through a walking gait.

In a walking gait, three legs are consistently on the ground. Figure 40 was a simulation of torques as the robot walked straight. This simulation was with a payload of approximately 3o pounds and the number that was most useful to our study was the max torque the 2nd joint peaking slightly over 200Nm. This was a huge factor in our motor selection, discussed in Section 3.2.

3.2 Motor and Gearbox Selection

Using the torque outputs of the MATLAB simulations, the team could determine the specifications of the motors needed to support the robot with a payload. The process for selecting the motors is outlined in this section.

3.2.1 Hip Motor Selection

As we completed our initial design we recognized that the hip motor would need to be selected independently of the knee and shoulder, due a larger torque exerted by the leg on at this point. Our current hip design calls for a custom designed linear actuator based off a lead screw and nut system. The team evaluated multiple different motor and lead screw options and settled on a ½-10" lead screw coupled with a brushless Maxon EC-max 40 motor and GP 42 26:1 gearbox. For the sake of being concise, this section will explain the calculations completed for just this one motor rather than going through each iteration separately.

First the team would need to define the various constants that we have. These include the lead screw reduction, the torque of the motor, the efficiency of the system, the gearbox reduction, the external reduction, the motor RPM and the lever arm on the system. They can be seen defined as shown here:

$$\text{Screw}_{\text{turn}}\text{= 0.1in}$$
$$\text{Torque}_{\text{motor}} = 0.0885\text{N*m}$$
$$\text{Efficiency=0.9}$$
$$\text{Gear}_{\text{planetary}} = 0.9$$
$$\text{Gear}_{\text{external}} = 1$$
$$\text{Motor}_{\text{nominal}} = 10200 \text{ rpm}$$
$$\text{Lever}_{\text{arm}} = 3.25\text{in}$$

Next the team calculated the force exerted on the system by the lead screw. Additionally, using the lever arm from the linear actuator to the pivot point we can also calculate the torque which the motor puts out on the system. This can be computed as shown in Equation 1 and Equation 2.

$$F_{leadscrew} = \frac{2 * \pi * Torque_{motor} * Gear_{planetary} * Gear_{external} * Efficiency}{Screw_{turn}} = 5123\ N$$

Equation 1 - Force from leadscrew

$$Torque_{leadscrew} = F_{leadscrew} * Lever_{arm} = 423N * m$$

Equation 2 - Torque from leadscrew

Now, knowing the theoretical torque which the motor can output we must see how much torque the hip system requires. In this case we conducted a simple worst case static analysis assuming the robot is on 3 legs shown in Equation 3.

$$Torque_{leg} = \frac{30in * 300lbf}{3} = 338.9N * m$$

Equation 3 - Torque required on leg

Finally, we can check to make sure that this torque can in fact be outputted by our motors by checking the safety factor. This was also completed as shown in Equation 4.

$$SafetyFactor = \frac{Torque_{leadscrew}}{Torque_{leg}} = 1.248$$

Equation 4 - Safety Factor of hip motor

As can be seen the motors will clearly have enough torque to move the legs even in a worst case scenario. However, the question still stands as to whether the motors will respond to our demands fast enough. Proceeding to look at the motor speed and factoring in the reduction of the planetary gearbox, the external gearbox and the lead screw we can find that the linear actuator will displace 0.65 inches every second. Knowing the lever arm and using some simple trigonometry we can further equate this to a displacement of the leg's bottom most point traversing 6 inches every second. While this may not be most ideal for a fully dynamically stable system this is not the top requirement of the system. For cases of simply turning the system and recovering from slight imbalances this will be plenty of speed.

3.2.2 Knee and Shoulder Motor Selection

Recognizing that in a worst case dynamic scenario we would see a worst case scenario power need from our shoulder and knee motors of 80rpm at 170N*m. This would equate to a total required power of 1.91hp. This would let us know that we will need to search for a motor which can output at least 2hp of power, fit within our allowed space (roughly 3" outer diameter by 10 inches long), have an easy gearbox solution, be highly efficient and still fit within our budget. These requirements limited our motor selection to a very narrow list. After contacting a few companies we settled on a few viable solutions which are outlined in Table 13. It should be noted that while we did look into stepper motors, the torque requirements relative to the motor size constraints would not be feasible.

	Maxon RE 50	Ampflow A28-150	Leeson M1120046	NPC-T64
Nominal Voltage (V)	24	24	24	24
Nominal Speed (rpm)	5538	6000	4300	230
Stall Torque (ozf*in)	1192	1970	1782	1320
Efficiency (%)	93	82	83	N/A
Total Power (W)	200	2237	1417	522
Max Amp Draw (A)	212	285	240	110
Weight (lb)	2.425	3.8	8.0	13
Cost ($)	450	270	276	331
Lead Time	12 weeks	4 weeks	None	None

Table 13 - Prospective Motor Solutions

Looking at these motors performance values, we quickly determined that there were only two viable solutions – the Ampflow motors and the Maxon motors. The Leeson motor as well as the NPC Motor were quickly ruled out due to their size and weight. Additionally, we found that the NPC would be underpowered for our need while the Leeson would not have a simple gearbox solution. Consequently, there were only two options. Looking into the two remaining choses, the Maxon sported a lower weight and higher efficiency while delivering almost the same amount of speed. However, the price was significantly higher as was the lead time. Additionally, the stall torque of the Maxon was significantly lower than the Ampflow motor. Given the disparity in cost and extensive lead time, we chose to go with the Ampflow motors.

3.2.3 Knee and Shoulder Gearbox Selection

Understanding the speed requirements from our motors we began to look for gearboxes to drastically reduce the max speed of the motors. As can be seen in Figure 41, motors develop a very low overall efficiency at low speeds. Given that our motors have a nominal speed of 6000RPM and our system calls for roughly 200RPM the team would need to explore options to gear down the system to bring the overall efficiency up.

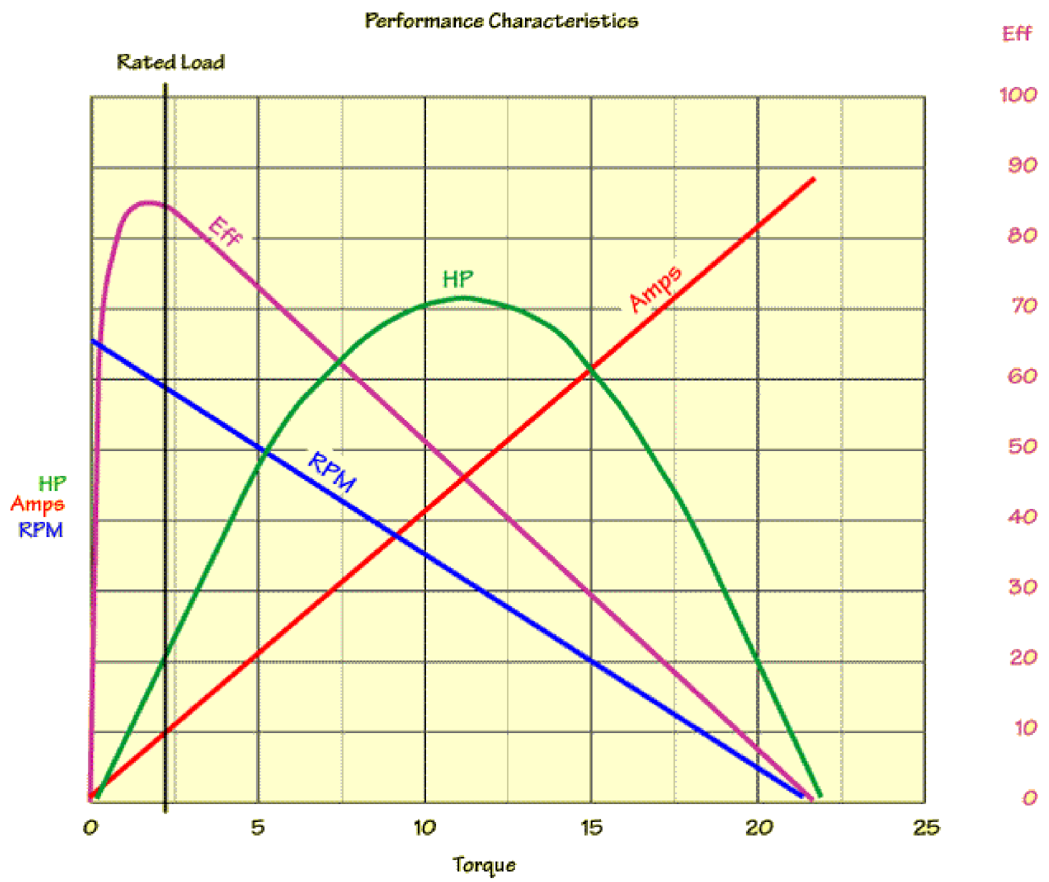


Figure 41 - Performance characteristics of a typical DC motor

There were two viable options to bring the overall speed of the motor down: a gear-down through a cable system with pulleys or a gear-down through a simple gearbox. The team was able to first eliminate the idea of using a cable system to gear down the motor. Using a cable system to gear down the motors would severely increase the complexity and weight of the system. Additionally, the system would require for cables to come out of the plane of the system. Finally, any cable gear-down system would need to be custom designed and manufactured. The extreme complexity in manufacturing this system could not be justified when compared to a simple gearbox. We then began to look into various gearbox alternatives. Knowing that we needed to get to roughly 200 RPM and knowing that we were starting with a nominal speed of 6000 RPM this called for roughly a 30:1 reduction. The team explored a number of different gearbox options including planetary gearboxes, helical planetary gearboxes, harmonic gearboxes and cycloidal gearboxes. Due to the extreme precision, low backlash required, time constrains and high amount of torque needed for this system we did not consider manufacturing our own gearboxes to be a viable solution. The gearbox would need to be roughly a 30:1 reduction, fit our Ampflow A28-150 motor, be less than 5.9 inches long, and work for at least 150N*m of torque. A comparison of all the explored commercial gearboxes can be seen in Table 14.

Brand	Gearbox Type	Reduction	Rated Torque	Backlash	Weight (lb)	Cost (each)	Lead-Time
Candy AE070	Harmonic	25:1	180 N*m	<12 arcmin	3.5	$700	2 weeks
Wittenstein LP120	Planetary	30:1	190N*m	<15 arcmin	11.1	$1327	2 weeks
Onvio AH-030	Helical Planetary	25:1	293N*m	<6 arcmin	18.7	$1185	4-6 weeks
Onvio M03	Cycloidal	25:1	225N*m	0 arcmin	8.8	$1528	6-8 weeks
Onvio AP-020	Planetary	28:1	140N*m	<8arcmin	7.5	$500	4 weeks
Apex AE-070	Helical Planetary	30:1	180N*m	<8arcmin	3.5	$467	1-2 weeks

Table 14 - Explored commercial gearbox options

Seeing these options we were able to immediately eliminate some of the options based on cost. Specifically, the Wittenstein LP120, Onvio AH-030, and Onvio M03 were outside of our price range. This left us with the Candy AE070, Onvio AP-020 and Apex AE-070. The Apex AE-070 appeared to be the best choice based on all the parameters; however, they do not work without heavy modification needed from the motors. Considering that the Candy AE070 gearbox would cost a total of $1600 more with little benefit the team made the choice to go with the Onvio AP-020 gearbox. The AP-020 gearbox also had the additional added benefit of having lower backlash. While our original parameters did call for at least 150N*m of torque, the team must take great care understanding the cycle time which these gearboxes are rated for. The 140N*m of torque which the gearbox is rated for is for a 100,000 life time. Consequently, the team's application will never see this cycle time and for short periods of time this extreme torque will not negatively affect the system.

3.3 Mechanical Systems

Each mechanical system went through several iterations before arriving at their final design. Though the final solution achieved may seem obvious, they are the product of in depth analysis.

3.3.1 Hip Subsystem

The hip of the robot is the revolute joint perpendicular to the knee and shoulder joints that allow the legs to camber in and out from the body. This functionality is needed not only for foot placement on uneven terrain, but also for stabilization when the robot is off balance and for sidestepping and turning. The hip system not only creates this motion, but is also the interface between the legs and main body of the robot. It was determined through examination of BigDog that for the hip to keep the robot stable it would require approximately ±15° of camber at a speed of about 10° per second.

The first option for generating this motion would be to put a rotational input directly on the hip joint, such as driving it with a motor and gearing. While this is the most direct approach, it was not chosen for a number of reasons. The system would have packaging issues fitting into the size requirements for the robot and most importantly it would be backdriveable. If the leg was forced out of place during operation the robot may become unstable and fall. While this issue could have been solved, it would be at the cost of another layer of complexity and cost.

The solution chosen was the concept shown in Figure 42. Torque for the hip joint would be generated by creating an offset linear force with a linear actuator. As there would only be a few degrees of travel, transition angles can be considered nearly ideal for the full range of motion.

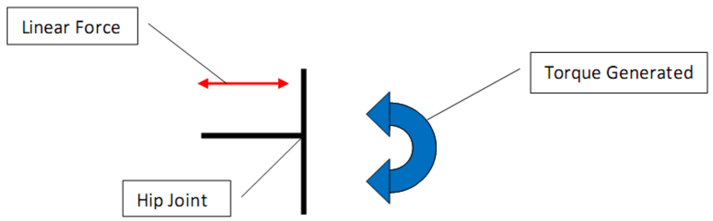


Figure 42 - Hip Actuation Concept

The design focus then shifted to selection or design of a linear actuator. Several off the shelf and ready to go items included pneumatic and hydraulic cylinders, but these choices were quickly ruled out. Pneumatics would have no intermediary positions and would be able to be force out of place, and either system would require the addition of a secondary power system such as an accumulator and pump. These systems would take up valuable weight and space in the robot.

Therefore the most reasonable option for the linear actuator would be a motor driven actuator. These types or actuators are also commercially available, but after research into what was available the team determined that a suitable actuator could not be purchased. Actuators rated for the required linear force were generally too large to fit within the size constraints of the robot, and custom made actuators were prohibitively expensive.

This meant that the team had to design and manufacture their own linear actuators. Lead screws were chosen as the actuators driver, just like the off the shelf counterparts. This was done because lead screws were the most elegant solution for the situation: they can generate linear force from torque; they produce a large reduction in a small space, and are highly backdrive resistant.

Figure 43 shows the first iteration of the linear actuator. A motor drives a lead screw held between two bearings with a nut riding on the screw. Push rods come out of the nut to not only stop it from rotating with the screw, but to transfer the linear motion from the nut to the end of the actuator. This was required to transfer the force around the bearing supporting the lead screw. The bearings supporting the lead screws cannot be ordinary radial ball bearings, as there are large axial forces generated by the lead screw, and transferred to the lead screw from lateral forces on the leg. For this reason angular contact bearings were used as they are intended to handle the combination loads exerted on the lead screw bearings.

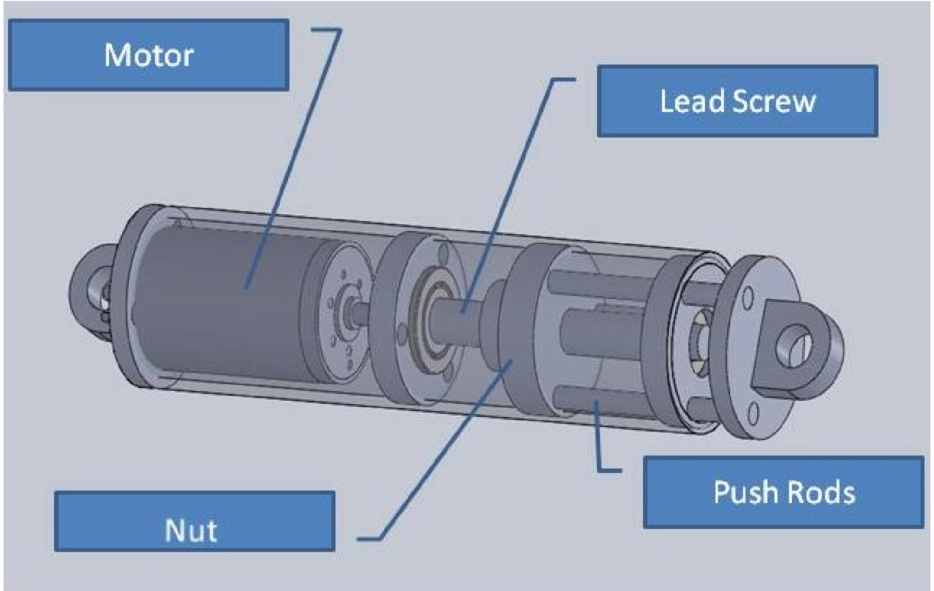


Figure 43- First Iteration of Linear Actuator

This initial iteration though was still too long to fit as desired in the frame, and required a 45° angle to fit correctly. The next and final iteration as seen in Figure 44 solves this problem by stacking the motor on top of the lead screw. Power is transferred from the motor by a 1:1 gear pair. Additionally, the bearing surface of the push rods was increased to ensure smooth movement.

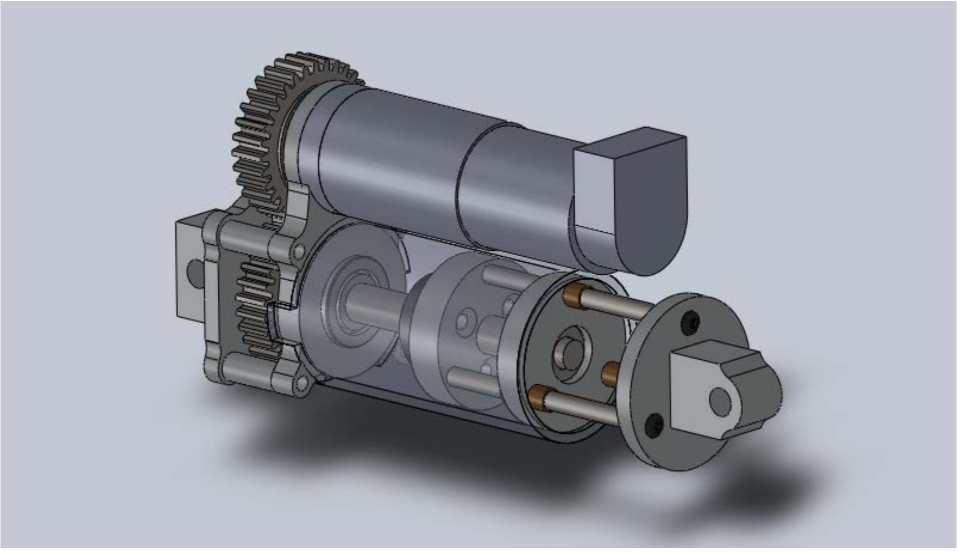


Figure 44- Final Actuator Design

The final part of the hip was the integration of the leg to the main body. This was done by making one pivot point that attached to the frame that would bear the load of the leg and then a second attachment point for the linear actuator. The first iteration as seen in Figure 45 shows the plus sign shape pivot that attaches to the leg plate, with a clevis above it to mount the linear actuator to the plate as well.

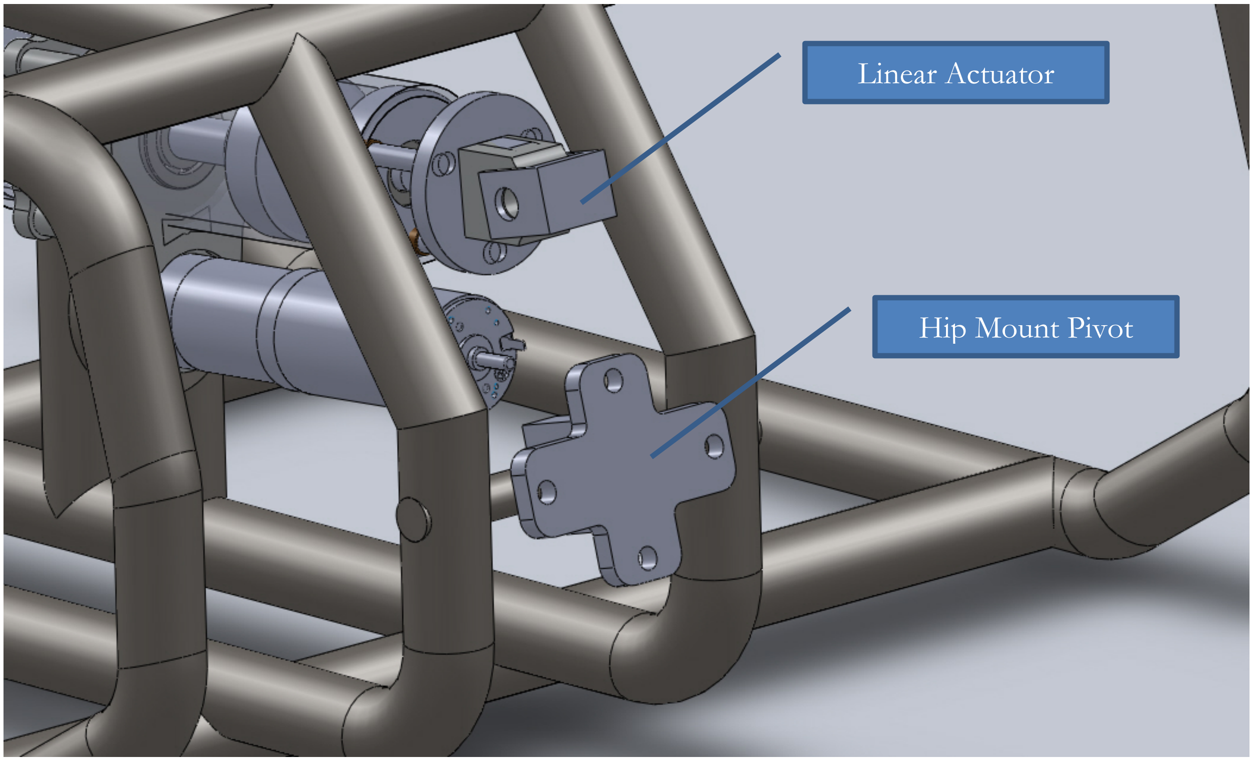


Figure 45- First Iteration Hip Mount

This was redesigned to the final iteration seen in Figure 46. Main improvement was the addition of the bearing blocks on either side of the hip mount shaft. This is to increase the strength of the hip mount from the thin walled steel tube to half an inch of aluminum. It also removed the problem of accurately placing and welding the mounting holes in the frame. The holes in the frame were kept only as clearance for easy assembly, allowing the shaft to slide in and out of the blocks. Additionally, mounting points were attached to install limit switches to stop the leg from over traveling.

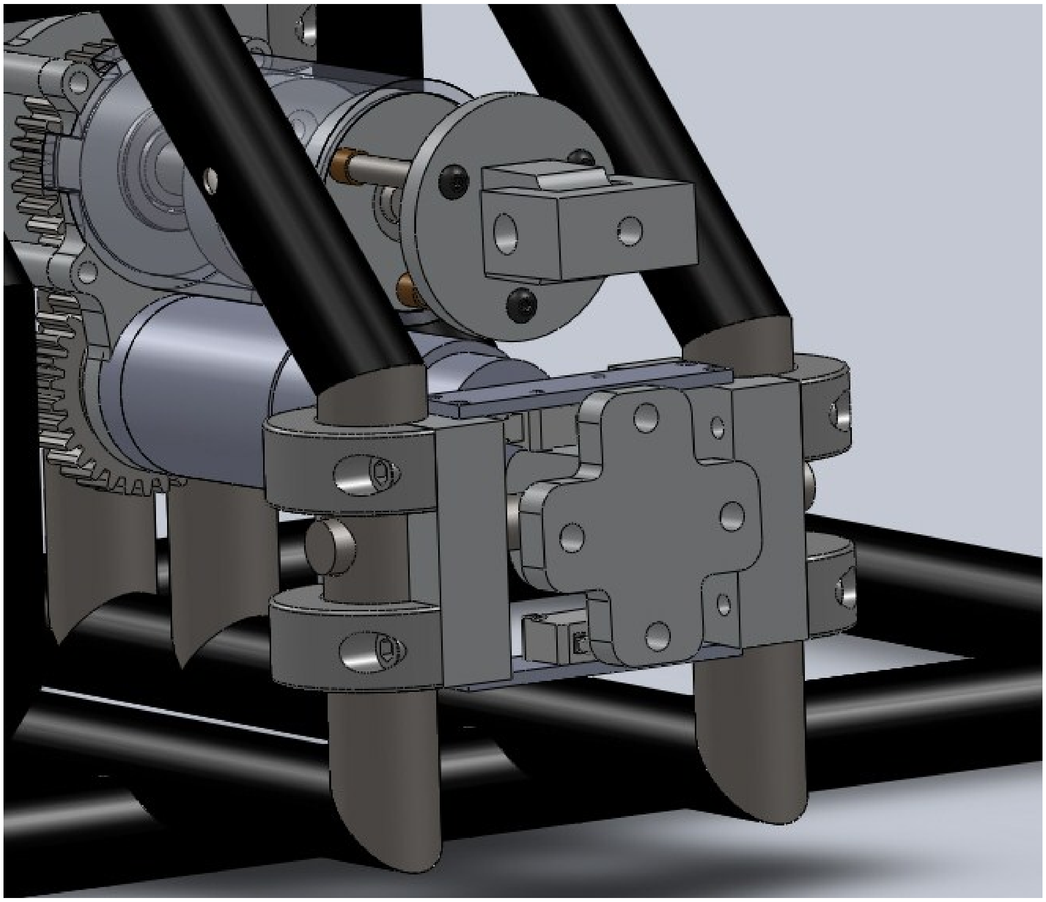


Figure 46- Hip Mount Final Iteration

No prototype of the linear actuator was made because of its straight forward design, and that to build a prototype would require most of the same parts and materials as the actual. Instead the actuator was the first system to be machined, assembled and tested. One actuator was assembled and tested before the other three were created.

The actuators were tested by running the actuator at full speed and timing the travel time to compare to what was calculated as the theoretical speed. Current was also monitored during the test, because jumps in current would denote load on the system caused by unacceptable friction.

3.3.2 Body Joint Subsystem

Designing the body joint subsystem began by first looking at the design requirements for the body joint. The body joint needed to mimic a quadrupedal animal spine by having a 2 degree of freedom highly compliant joint which would separate the front half of the body from the back half. The decision to implement this design requirement came for two reasons. First, the team wanted to proceed with their hope of achieving a biologically inspired system with the spine being shown to be a vital part of every animal's locomotion (Gray, 1932). Second, a fully compliant spine is a biological concept which has never been reproduced in a quadrupedal robotic system. We were interested in studying and seeing the effect of introducing a fully compliant spine to such a system. We decided to simplify the system and make the spine be a fully passive system but still have a bias to center the spine. While quadrupedal animals do use their tendons to move their spine we decided to go with the more simple passive system. However, we did make it a necessity to design the system with ideas in mind as to how to make it fully active. The idea would be to first study a passive system and if found that the dynamic system does not respond in the appropriate manner then we could move to either a fixed spine or an active spine.

Given these design requirements of a center biased 2 degree of freedom spine, the team's first step was looking at a previous Worcester Polytechnic Institute project entitled "*Design of a Roof Inspection Robot*". This project employed a two degree of freedom central body joint in a much smaller scale robotic system. The joint was actively controlled using a small DC motor through a sprocket gear-down. The original design for the central spine was heavily influenced by this project. The initial prototype design can be seen in Figure 47. The initial design, however, suffered from serious issues which were otherwise not a problem for the roof robot. In order to rotate the robot in one of the axes, the robot would need to revolve around the threads of a screw. While not being a serious issue in a small scale robot, having 150 pounds over a 1 foot moment arm in single shear would likely shear the screw and is generally bad engineering practice.

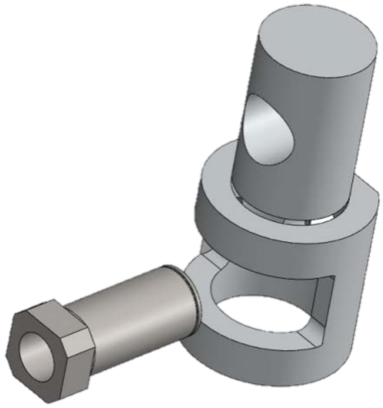


Figure 47 - Original Body Joint System Design

The team then moved on to their next iteration, the "car yoke" design. This design was modeled after a typical car yoke Car yokes are commonly used as steering mechanisms in cars to allow for a two degree of freedom steering mechanism in a compact size. Originally, we researched possible Commercial off the Shelf (COTS) solutions to our body joint issue; however, typical car yokes are made of bulky cast material which is not an ideal choice for our application. Given the strict weight requirements, we needed to have the most compact solution which would still achieve our desired objective. Using the same principles as a car steering mechanism, we proceeded to design our own universal joint (Figure 48). This mechanism relied on two 1.5" pins to hold the entire body in shear in two planes. The overall mechanism was quite bulky and had no method of directly attaching to the frame. Additionally, the main universal joint was a very large piece of material that would have been exceedingly difficult to manufacture out of a very expensive piece of stock.

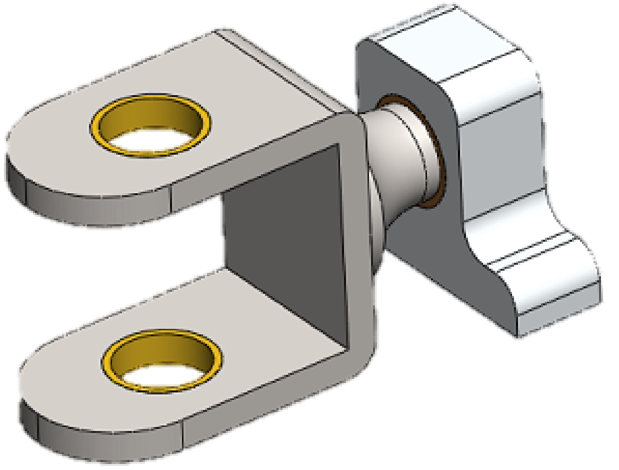


Figure 48 - "Car Yoke" Design

As such, the team proceeded to move on to their final iteration (Figure 49). This iteration managed to use COTS parts for both shafts in the form of large shoulder bolts. Two large bearing holders were designed and tested with ANSYS FEA tools to make sure they could withstand both the static and dynamic forces of the body. One of the many analyses conducted on the individual parts can be seen in Figure 50. This analysis was a simple deformation analysis to check that in a worst case dynamic scenario the body joint would not displace a crucial amount. In this case, the body joint would only move 0.004 mm, a perfectly acceptable number. Using this Finite Element Tools we were also able to redesign the structure to be a little bit beefier but made out of aluminum to significantly reduce the weight of the entire system. The system was also comprised of four gas springs mounted onto eight smaller universal joints. This would allow the spring mass dampeners to pivot with the body joint itself over a large range of motion. These gas springs were also commercially available in 10 pound force increments starting at 15 pounds and terminating at 150 pounds. The two shafts for the system were ¾" diameter and 1 ½" diameter. The reason for the disparity was to ensure that there was enough support on the shaft. While the ¾" diameter shaft was supported by 1 ¾" inches of material, the 1 ½" shaft was only supported by ¾" of material. Hence, it was imperative to choose a large bolt diameter to make sure that there was enough overall surface area supporting the shaft. The entire system attaches to the two halves of the round frame tubing with the use of multiple halves of shaft collars. On the body joint side there would be a groove cut for a 1" diameter tube such that the frame could sit halfway inside the body joint.

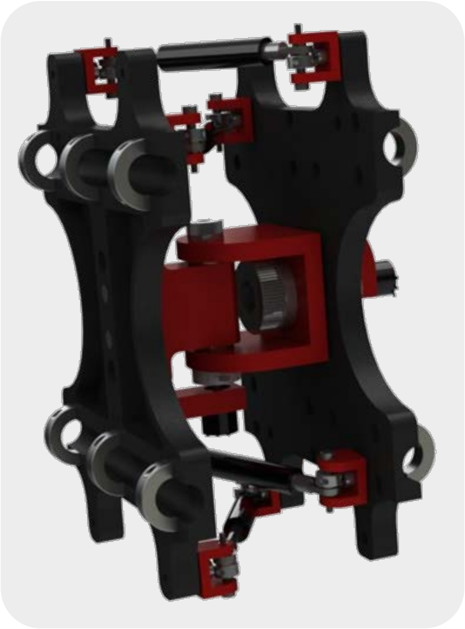


Figure 49 - Final Body Joint Render

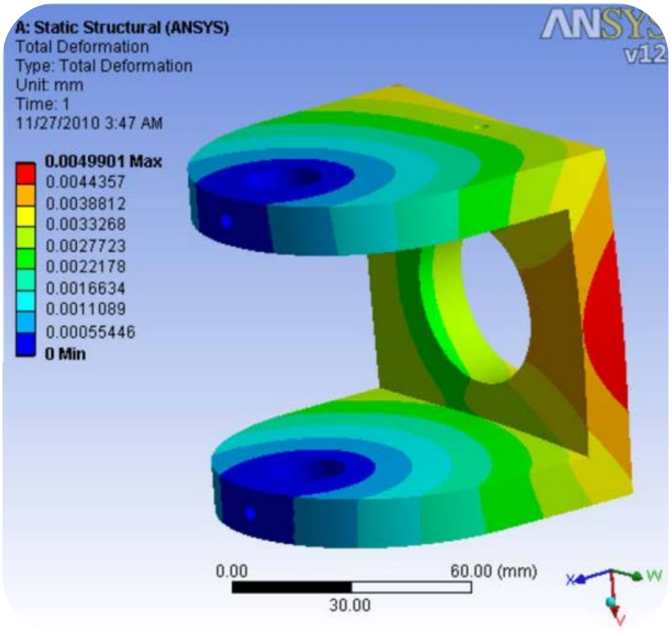


Figure 50 - Body Joint Deformation Analysis

Once fully designed, it became apparent that the system would first need to be prototyped. While SolidWorks was a great modeling tool, it proved to be difficult to use SolidWorks as a dynamic simulator to ensure proper operation of the device. While we were able to theoretically calculate that the system's maximum extents to be 50 degrees in the *roll* direction as well as 10 degrees in the *yaw* rotation we wanted to verify these calculations in a real life to scale prototype. Additionally, the prototype served to verify that that the springs would bias the body joint into the center position – something that is impossible to verify with a simple drafting tool. We proceeded to manufacture a full scale prototype out of wood, acrylic and ABS plastic. Parts were manufactured with the help of the laser cutter, bandsaw, drill-press and 3D Printer. Using this method we were able to very quickly test our system. While the tolerances associated with these materials were not ideal, we were able to verify that the body joint would have bias towards the center and could successfully achieve the proper amount of rotation. The body joint prototype also gave us a rough idea of the amount of force we would need from our gas springs. A picture of the final prototype can be seen in Figure 51.

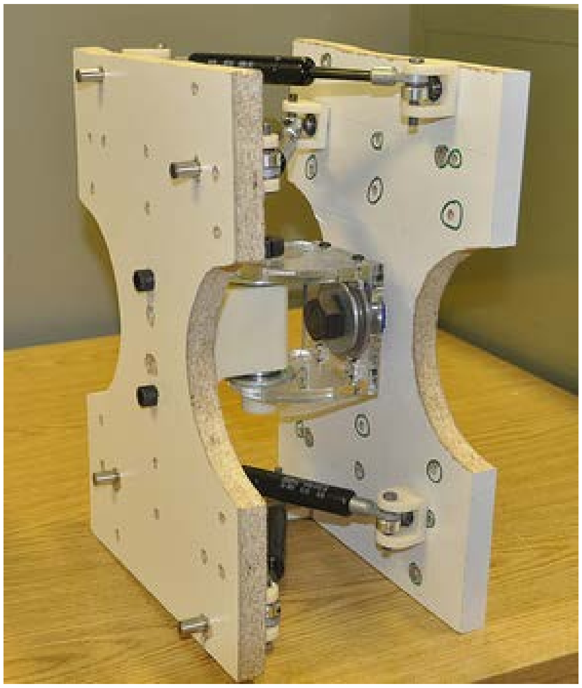


Figure 51 - Final Body Joint Prototype

3.3.3 Sensor Head Subsystem

The sensor head needed to be designed to house the on-board camera, the LIDAR, and two arrays of 80-lumen LEDs. Special care would be required to make sure the LIDAR is protected, since the LIDAR used is very expensive. However, the design would also need to be open enough for the LIDAR to be able to produce a full scan of the environment without interference from its housing. It was also important to design the sensor head to be capable of moving the camera and LIDAR without having to move the entire sensor head. To accomplish this, the head was designed to incorporate two servos to move the camera (for tilting and panning) and one servo for the LIDAR (for tilting). Figure 52Figure 53 is the initial design for the sensor head:

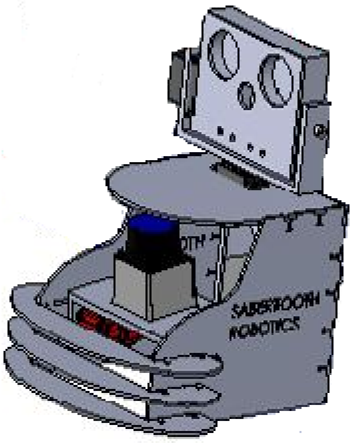


Figure 52 - Initial Sensor Head Design

From this initial design, a prototype was constructed from acrylic and aluminum. This was one of the first prototypes constructed to allow for software development with the LIDAR and camera to begin as soon as possible.

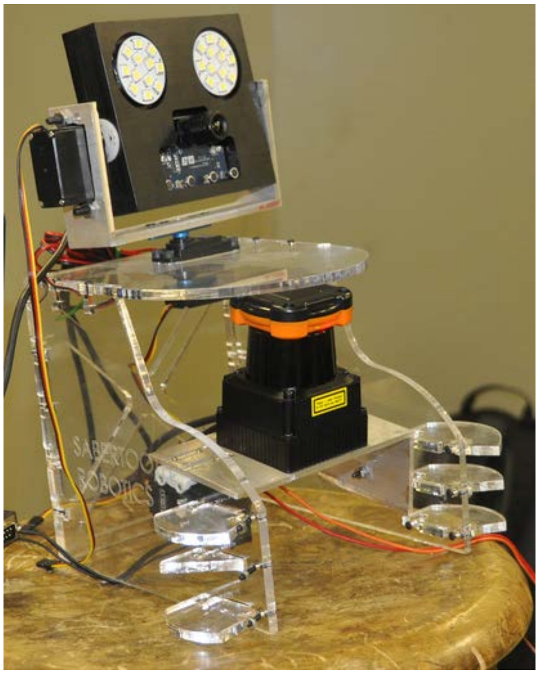


Figure 53 - Sensor Head Prototype

After many other aspects of the robot were redesigned and manufactured, the sensor head was also redesigned for aesthetic and functionality purposes. Figure 54 depicts the final design of the sensor head, constructed out of acrylic, aluminum, and acrylonitrile butadiene styrene (ABS):

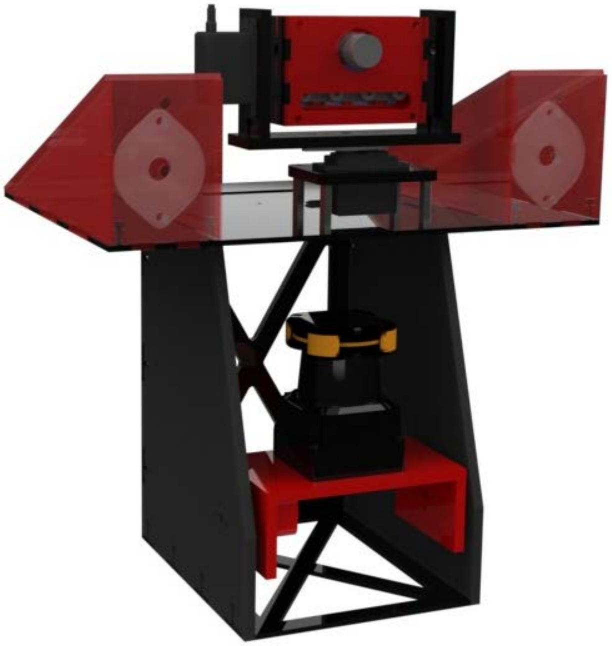


Figure 54 - Final Sensor Head Design

3.3.4 Body Frame Design

The frame of the main body of the robot serves several important functions. First, it is the main component to which the four legs are connected. Secondly, it is where all of the electronic and power systems are held to allow the robot to be mobile.

Because of the specification for a central body joint, the frame could not be constructed as one singular piece, but instead as two identical but smaller body sections. As mentioned in section 3.3.1 the hip system was also an integral part of the frame.

The first iteration of the body sections was constructed mainly of welded 1"x1", 1/8" wall aluminum box tubing, as seen in Figure 55. The outriders on the sides of the main section of the frame are mounting points for the hip pivots. The oval like shape of the frame was for ascetics, as a simple square frame looked awkward and unprofessional to many people when initially presented.

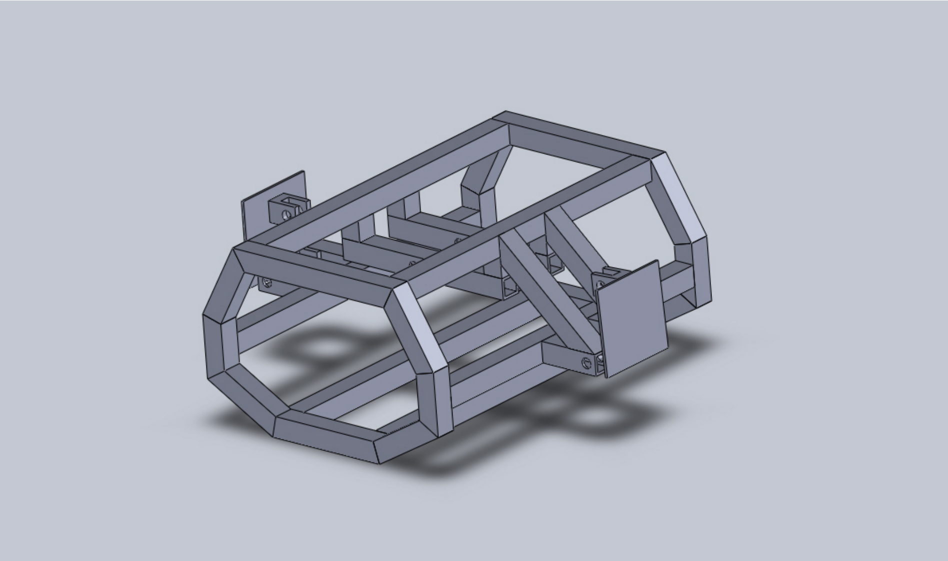


Figure 55- First Body Section Iteration

The team was still not satisfied with the frames appearance or how it had to be manufactured. It required the fabrication of many small pieces and hundreds of welds to assemble. The next iteration of the frame is shown in Figure 56. Thin walled round steel tubing was selected as the new frame material, as it was just as strong and light as the original box tubing, but could also be bent using a tube bender. The intent of this change was to reduce each frame section to less than 10 parts, and reduce the number of welds needed.

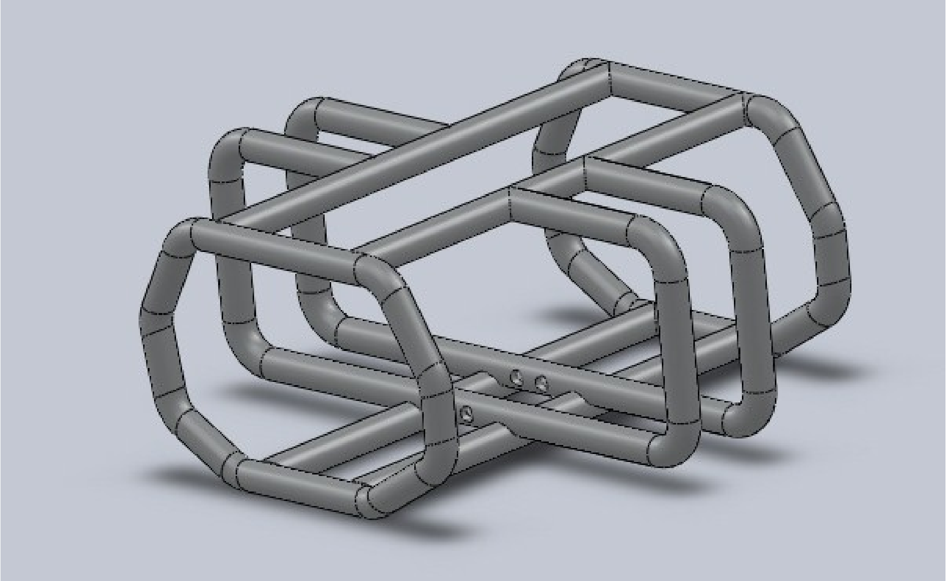


Figure 56- Frame Second Iteration

The final design, as seen in Figure 57, kept the round steel tubing but reverted back to the multi piece frame approach instead of bending. This was because of manufacturing difficulties involved with bending. Additionally the width of the main frame sections was extended out to the outriders to make the frame a uniform width. This greatly increased space for power systems.

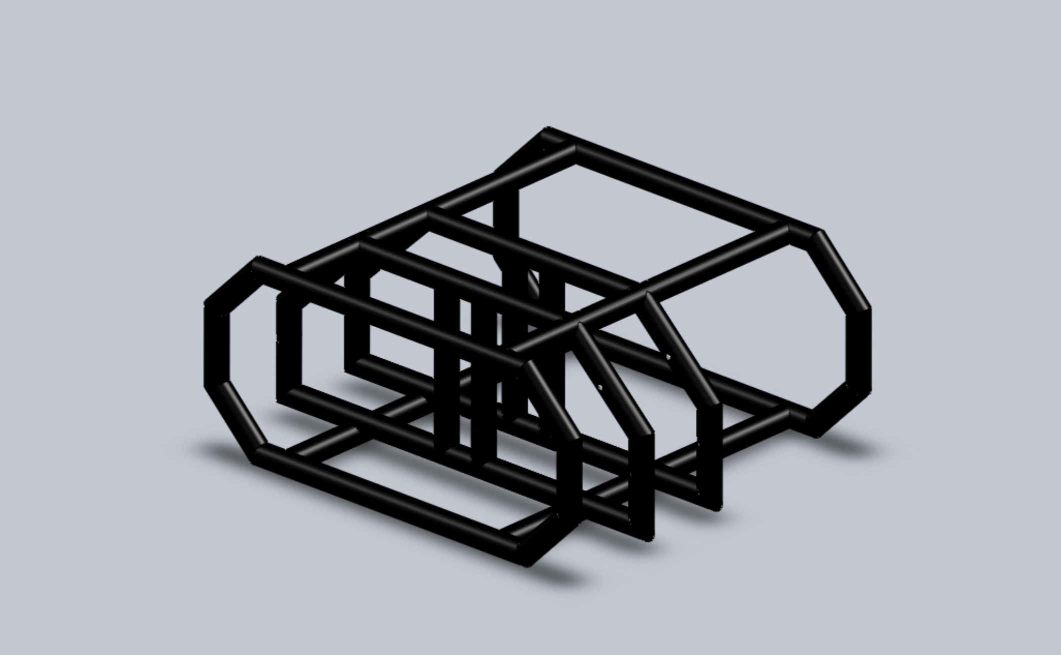


Figure 57- Final Body Section Design

FEA analysis, as seen in Figure 58, was performed on the bend piece design of the frame at the critical point of where the leg attaches to the frame. Even in the worst case scenario, the frame has a minimum safety factor of approximately 14, showing that the frame is more than sufficiently durable, even if the robot were to fall full force upon it.

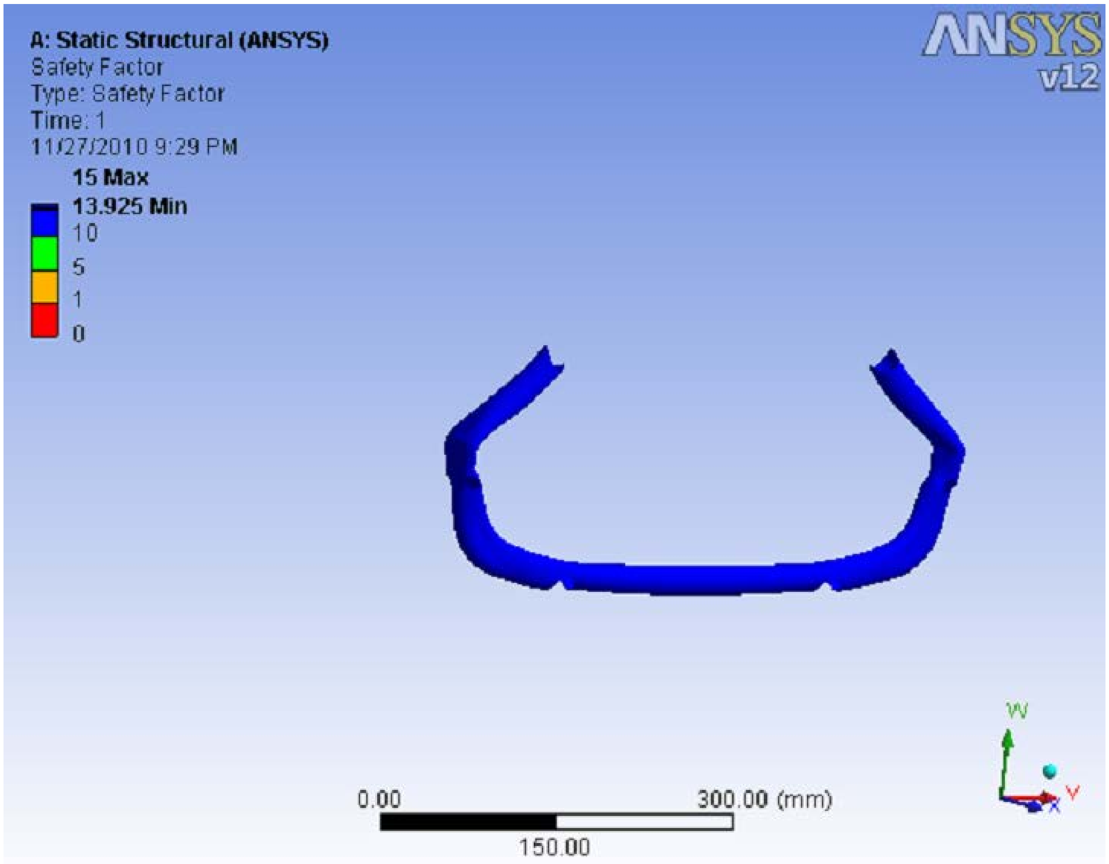


Figure 58- FEA on Hip Mount Frame

The final frame was constructed by cutting appropriate angle or notches on the ends of the tube sections on either a CNC machine or an abrasive saw. The frame pieces were then jigged into place and welded. Tolerances were held to within .01" for critical features such as mounting hole placement and overall width and height, while less critical dimensions were held to approximately .125" to allow for irregularities in fitting and welding.

3.3.5 Spring System

One of the team's primary design requirements included implementing the SLIP model into our design. Upon careful research of a number of complex robotic systems including the Raibert

Hopper, Spring Flamingo, and MABEL the team grasped that the most effective way to implement the SLIP model would be to introduce springs in series with the motor. The team shortly considered putting springs in parallel with the motor through a cam system but found on closer inspection that it would not produce the intended response. Additionally the parallel spring solution would only have a positive effect for one gait and instead incur additional energy losses through all others. As such, the team moved forward with the idea of implementing springs in series with the motors. Both the shoulder and knee motor would need springs in line with the system. A simplified picture depicting the team's implementation of the SLIP model in conjunction with the cable system can be seen in Figure 59.

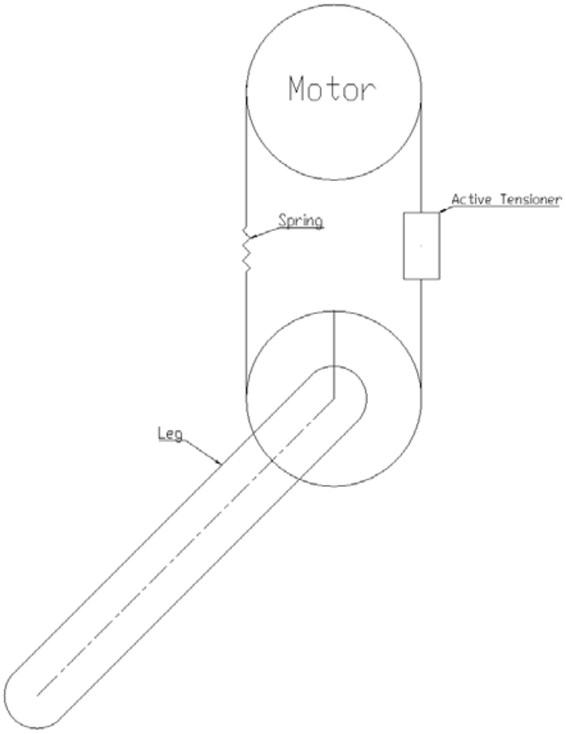


Figure 59 - Implemented SLIP Model

Taking the calculations for the motor torque, the team could actively calculate the amount of force that the cable would be undergoing. Knowing that the shoulder joint would see a worst case scenario torque of 210N*m and that the knee joint would see a worst case scenario torque of

150N*m. Exploring various alternatives the team first began by exploring the use of .125" wire rope. Trying to adhere to handbook tolerances the team would need to use capstons with a minimum of a 2" bend radius. Understanding the need of making all assemblies low profile the team proceeded to design around 2" radius capstons. The radius of the pulley could be divided by the total torque to find that the force in the shoulder motor cable would be 920lb and the force in the knee motor cable would be 660lb. Using these values the team could then look for appropriate springs to fit this criteria. An area of SLIP which has been explored with very little vigor is leg stiffness. While a number of springs will fit the general requirement of being able to take up 920lb of slack, the rotation of the leg will differ depending on the spring's constant. As a starting point, the team specified springs which aimed to rotate the leg 30 degrees under full spring deflection. This number was chosen based off some of the preliminary research completed on variable spring stiffness on the MABEL platform (Hurst, 2008). However, the team would make it a functional requirement to be able and easily swap the springs to change the leg stiffness. This would equate to a deflection of roughly 1 inch in both springs as explained in Equation 5 based off the formula for the circumference of a circle.

$$Spring\ Deflection = 2 * \pi * 2in * \frac{30°}{360°} = 1.047\ in$$

Equation 5 - Calculated Spring Deflection

From here the team could determine that they would need to select springs which would have a spring constant of roughly 920lb/in for the shoulder motor and 660lb/in for the knee motor. Upon investigating extension springs the team found that it was not feasible to select springs which would fit into a compact form. Upon further research, the team found that compression springs have a much higher energy density (Lee Spring Company, 2011) and should be explored as an alternative option to extension springs. As such another mechanism would need to be designed for the purpose of attaching compression springs and making them mimic the functional capabilities of extension springs. First, the team began by selecting a set of compression springs to achieve the required stiffness parameters previously defined. Springs with a large wire diameter, short length, acceptable rod diameter and spring deflection rate were preferred. After much searching the team selected two springs of 1000lb/in of deflection for the shoulder spring and 735lb/in for the knee spring.

Next came the task of designing a mechanism to make the compression springs effectively work the same as extension springs. The following was developed by using a "spring pod" design (Figure 60).

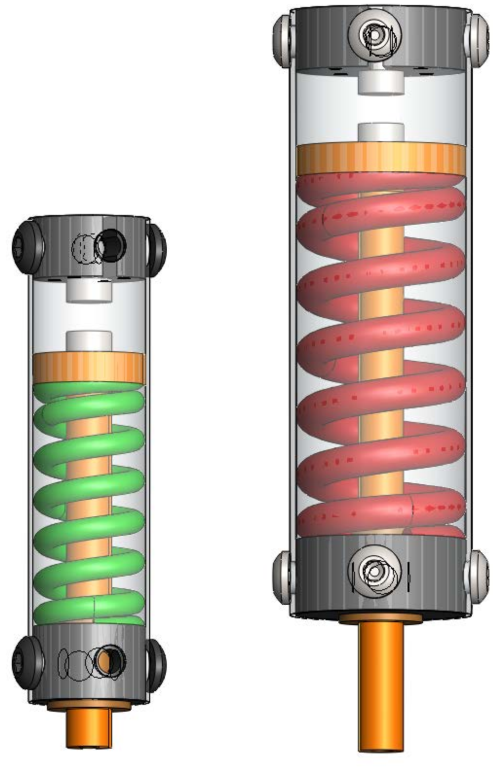


Figure 60 - Knee and Shoulder "Spring Pod" pictured left to right

The "spring pod" design employs 5 key elements: the plunger, the spring, the 2 hard stops, the tube and the ball and shank cable connector. In this design, the clear tube is designed out of aluminum to be the ground element of the system. The 2 hard stops attach to the aluminum tube using short ¼-20 screws. They are designed out of steel knowing that they will be taking constant cyclic load from the springs and ball and shank connectors. The plunger (pictured orange) pushes on the spring when the cable attached to the ball and shank connector pulls on the plunger. It should be noted that the cable runs through the center of the plunger. The hard stops are constantly taking loading from the two ball and shank connectors. Using this system, the robot is able to store energy in an extremely compact form.

3.3.6 Cable System

CABLE SELECTION

The cable system would prove to be one of the most complex. The system would need to be designed such that energy could be successfully transferred from the motors, through the springs and tensioners, and into the pivoting joint. First the proper form of wire rope would need to be selected for the system. Understanding that a capston system with round drums would be used it was determined that the team would use .125in 7x19 construction wire rope. 7 x 19 wire rope is able to easily bend around drums as well as have high yield strength of at least 2000 pounds. Upon further research, the team was able to find that wire rope is commonly very stretchy and that the team may run into issues with lack of proper tensioning due to the extreme amount of force on the cable (Lexco Cable, 2011). The team proceeded to test the cable and see if it would be feasible to bring the cable into the plastic region hence permanently deforming it and not having to worry about any of the possible stretching issues. For this task, the team would need to find a way to pull the cable with upwards of 2000 pounds of force and record the displacement as well as the force set on the cable. The Instron tensile tester in Worcester Polytechnic Institute's Civil Engineering Department proved to be the ideal solution as it could easily pull with the required amount of force to break the cable and also accurately record back data. The team developed a custom fixture for this machine using three sets of wire cable clamps (Figure 61).

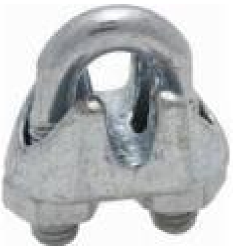


Figure 61 - Wire Cable Clamp

Upon pulling the cable the team was able to receive a full force displacement curve as shown in Figure 62. Using this data it can clearly be concluded that pre-stretching the cable is not a valid solution to our issue since the cable never falls into the plastic region like most other materials. Instead, the team proceeded to look into using tensioners with a high rate of adjustability.

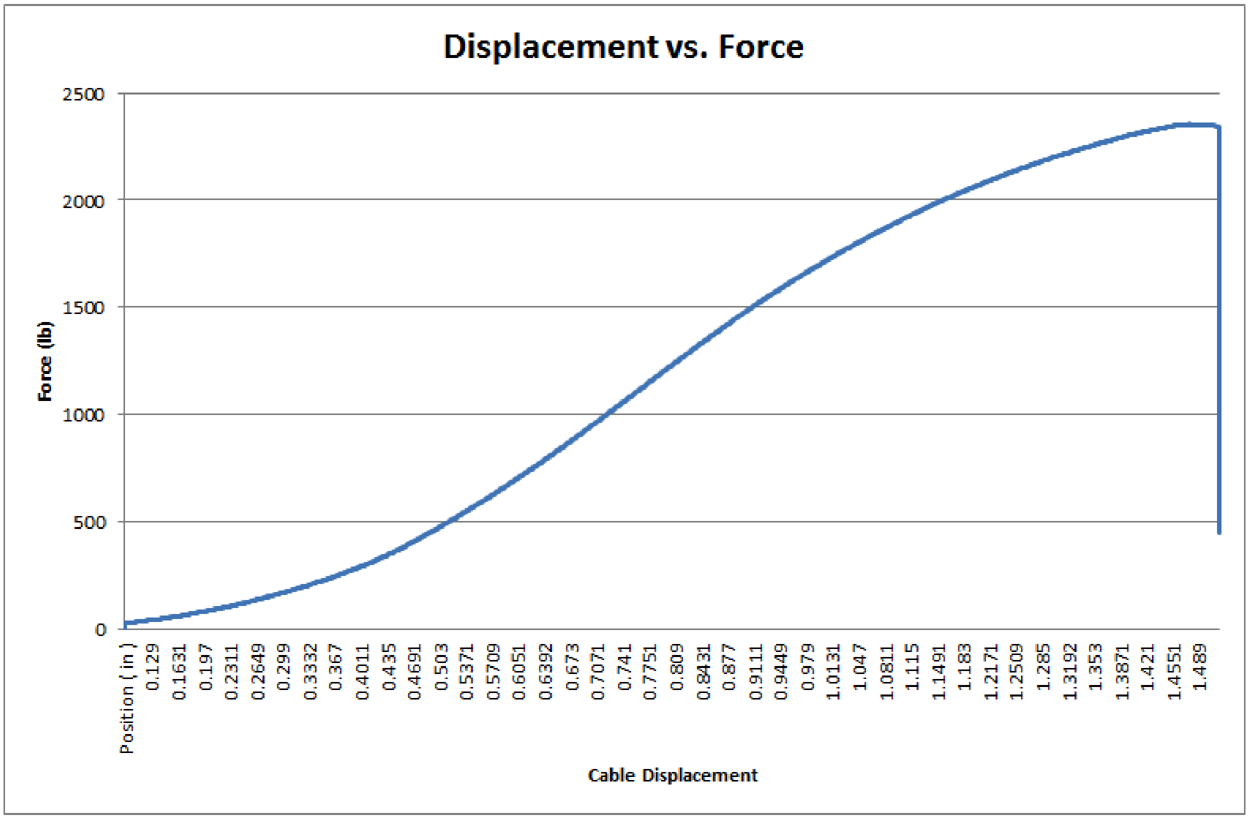


Figure 62 - Cable Displacement

CABLE TENSIONERS

The team then proceeded to design a cable tensioning system that would be low profile, low weight and overall mechanically simple. This was completed by using wire rope studs which had an adjustability of up to 0.5 inches. This adjustability would be triggered by tightening a nut on the end of the capston and in turn pulling the cable up with it. The tensioners had two extremes and by combining 2 capstons, the total cable adjustability would be up to 1 inch.

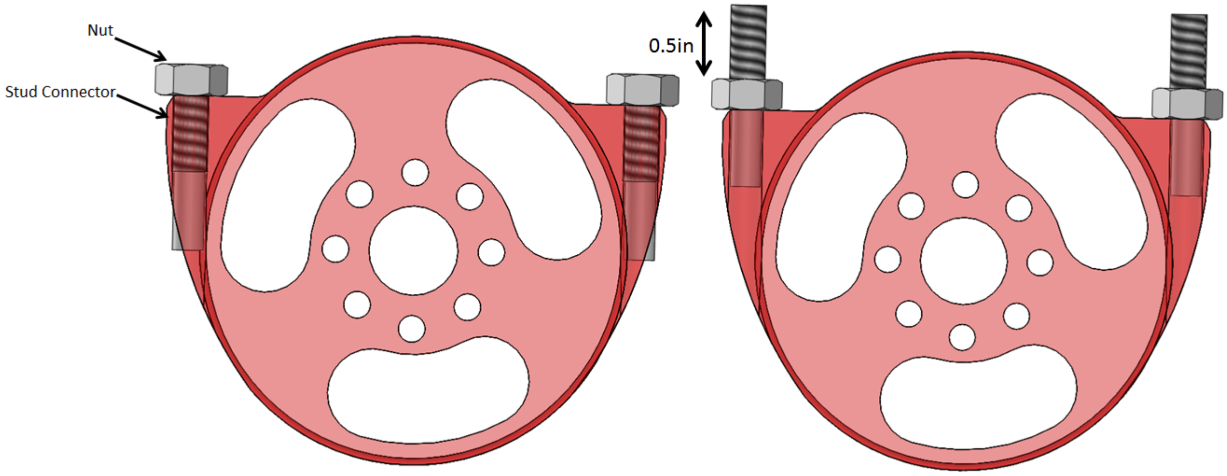


Figure 63 - Capston and Tensioner Design

In addition to the passive tensioner design the team also designed active tensioners to take any additional slack. This was especially necessary when trying to take up the extra slack on the opposite side of the compression springs. This design would need to actively take up up-to 1 inch of cable slack. The team began by inspecting other tensioner systems, specifically those on chains in bicycles. Figure 64 depicts two tensioner designs for a bicycle. The figure on the left shows a design employing an extension spring to constantly push out on the chain. Alternatively, the figure on the right depicts a system which uses a torsion spring to wrap the cable around and take up the necessary slack with that system.

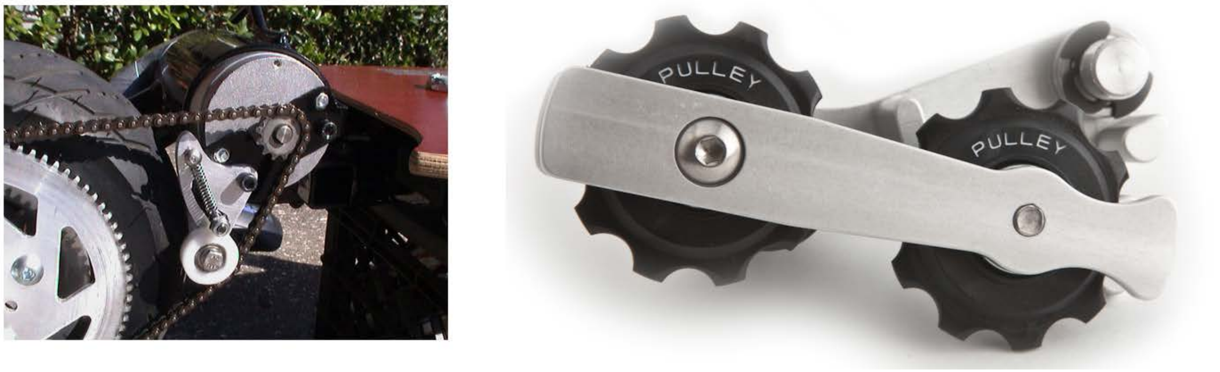


Figure 64 - Bicycle Tensioners

The figure on the right would serve to be the modeling base for our own tensioners. This was decided for two reasons. Our cable system has an extreme amount of complexity due to the large number of moving parts. The tensioner would need to have an extremely low profile to avoid any interference with other portions of the system. The extension spring would need to be very large to take up 1" of slack. Additionally, in both examples in Figure 64 the extension spring tensioner is pulling the cable off the sprocket. While this is only a small issue in sprockets, a cable system is much stronger affected by such a tensioner and has a high capability of derailing. Instead, a high priority of the team's design would be to make sure the tensioner pulled the cable tighter onto the capston. The proposed design picture in Figure 65 came after multiple iterations, however all stemmed from the same core concept of a torsion spring to pull on the cable. The major iterations involved bending the original torsion spring to allow for a smaller profile and using a commercially developed modified shaft collar as the ground of the system.

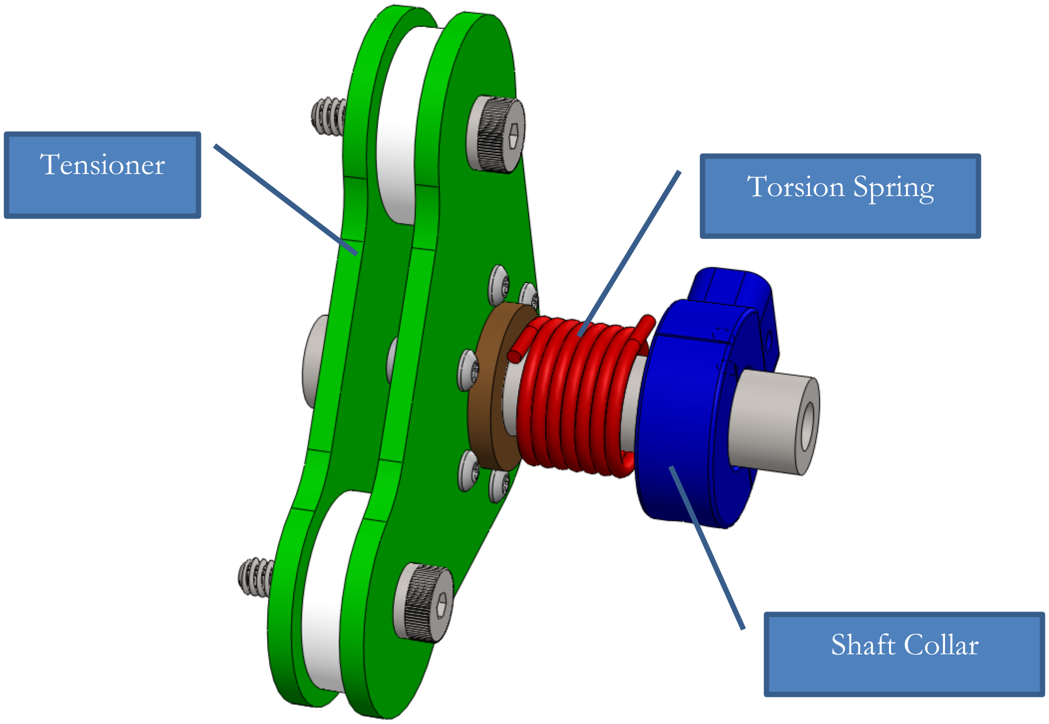


Figure 65 - Proposed Active Tensioner Design

In Figure 65, the blue element is the ground of the system developed in the form of a commercially modified shaft collar. The red element is a torsion spring. For the sake of drafting simplicity, out of plane bends on the torsion spring were not modeled. The two green elements attach to the ground element through the torsion spring and are preloaded to rotate the torsion spring 90 degrees in a position where all the compression springs are uncompressed. They rotate on the bronze bushing about the dead shaft. The cable is designed to go between the white rollers such that it is always getting pulled in to the center of the capston. The 10 ft*lb torsion spring was selected by picking a spring which would fit within the general size and still have a force which would be great enough to move the loose cable and tensioner assembly.

CABLE ROUTING

Finally, the team was tasked with fitting in all of the required components into an extremely compact form. The design would need to encompass capstons for the motors, springs in line with the motor, and capstons for both the shoulder and knee system. Additionally, the design called for an idler pulley at the shoulder to drive the knee. This was necessary so that when the shoulder was moving, the effective length to the knee was not affected. Finally, as the design evolved the team found it necessary to add a set of idler pulleys from both the knee and shoulder motors. This would allow for a more compact design. In order to accommodate the springs there would need to be at least twelve inches of travel. This was determined after carefully looking at the most extreme gait angles. In the most extreme gait, the robot would need to travel 90 degrees with the shoulder joint. This would mean that the cable travels the distance denoted by Equation 6 which takes into account the full leg travel as the possible spring expansion.

$$Cable\ Travel = 2 * \pi * 2\ in * \frac{90°}{360°} = 3.14\ in$$

Equation 6 - Cable Travel

Additionally, the team would also need to account for the total length of the expanded spring. The expanded spring pod itself was 6.5 inches long. In total that would mean that the tangent distances between the shoulder motor caspston and the shoulder driving capston would need to be 9.64in. Given the allowed location of the motors within the frame it quickly became clear that there would be an issue fitting all of the required pulleys into a small and compact form. Simply moving the motors to a location 9.6 inches away from the previous pulley would yield an extremely large and bulky shoulder plate. Instead, the team chose a secondary option of adding an additional set of idler pulleys, hence being able to inset the motors into small allocated portions of the frame. The downside of this solution, however, was the addition of 4 more idler pulleys per each leg. A full annotated CAD model of the shoulder plate design can be seen in Figure 66.

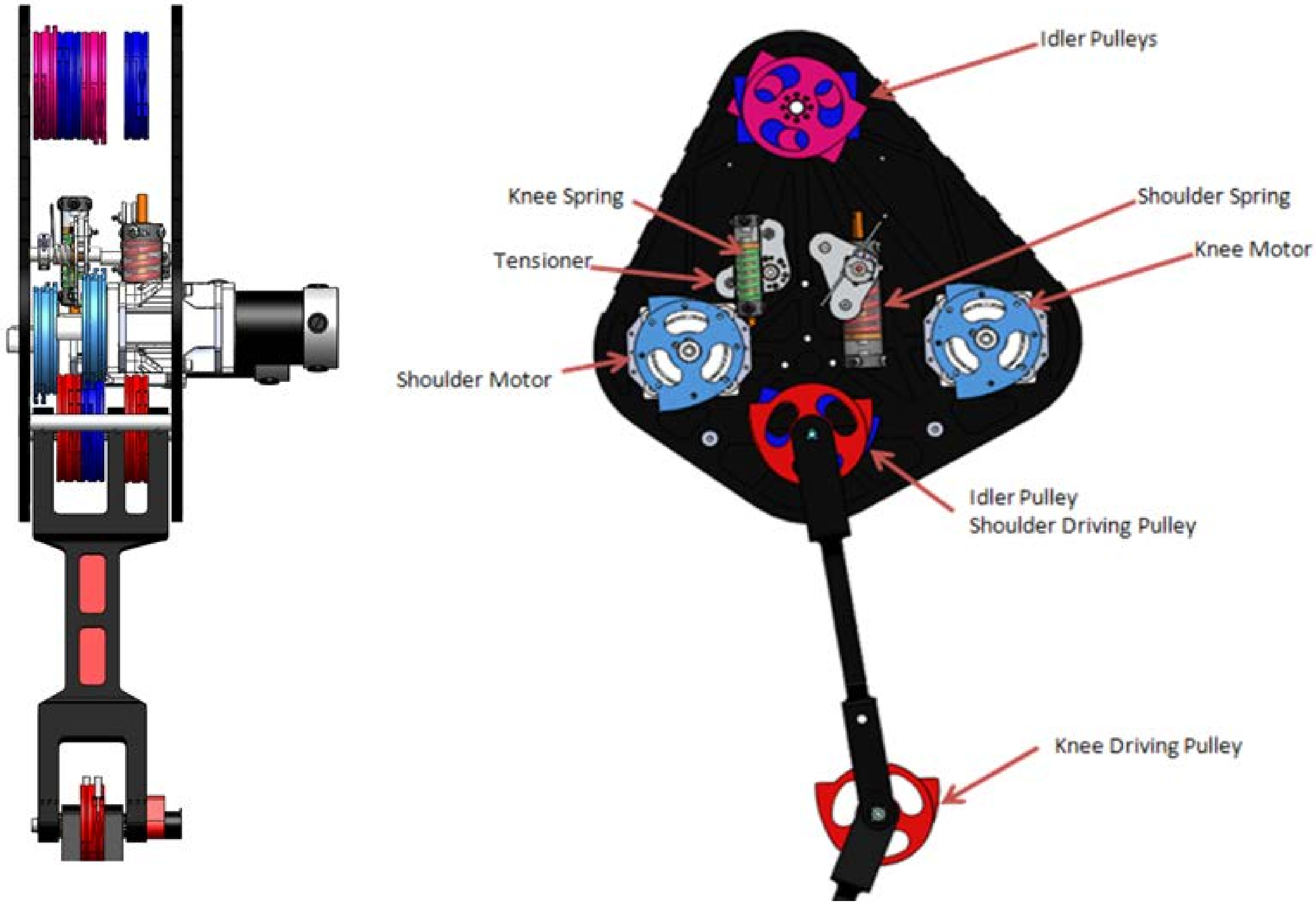


Figure 66 - CAD Model of Shoulder Plate Design

In this model, the motion of the system begins at the shoulder or knee motor pulleys (pictured light blue). This pulley then transmits power through a cable to the idler pulleys at the top of the shoulder plate (pictured pink). On the same shaft lies a second pulley (pictured dark blue) pinned with four 0.187" pins to the first. This pulley is designed to then take the power transmitted to the first idler pulley and retransmit it down through the rest of the system. This pulley then runs a cable down to the shoulder joint. In line with this cable is the spring pod module on one side and the active tensioner on the other. In the case of the shoulder joint, this is the end of the line and the final pulley (pictured red) is then pinned with four .187" pins to the shoulder. However, in the case of the knee motor there is one final step. The shoulder axis holds one more idler for the knee (pictured red). This idler is again pinned to a pulley on the same axis (pictured blue) and is then attached with cables to the final knee joint (pictured red).

It should also be noted that while all the pulleys may look identical that the four separate colors do correspond to significant changes. In order to make two mating pulleys work together they would need to be mirror images of one another such that their termination points ended up on opposite sides of one another (Figure 67).

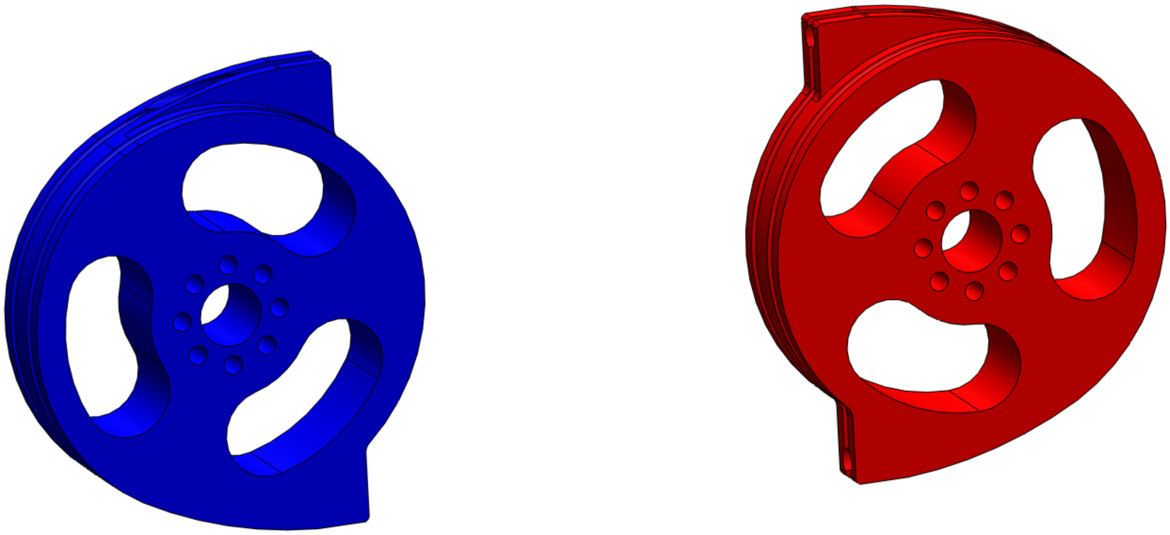


Figure 67 - Two of the four pulley configurations

Next, in order to accommodate for the top idlers having an angular offset relative to the previous joint the team would need to incorporate an offset bolt pattern in the design. This was necessary to ensure that there would be no lost rotation as a result of the offset motor locations. The two different bolt patterns can be seen in Figure 68.

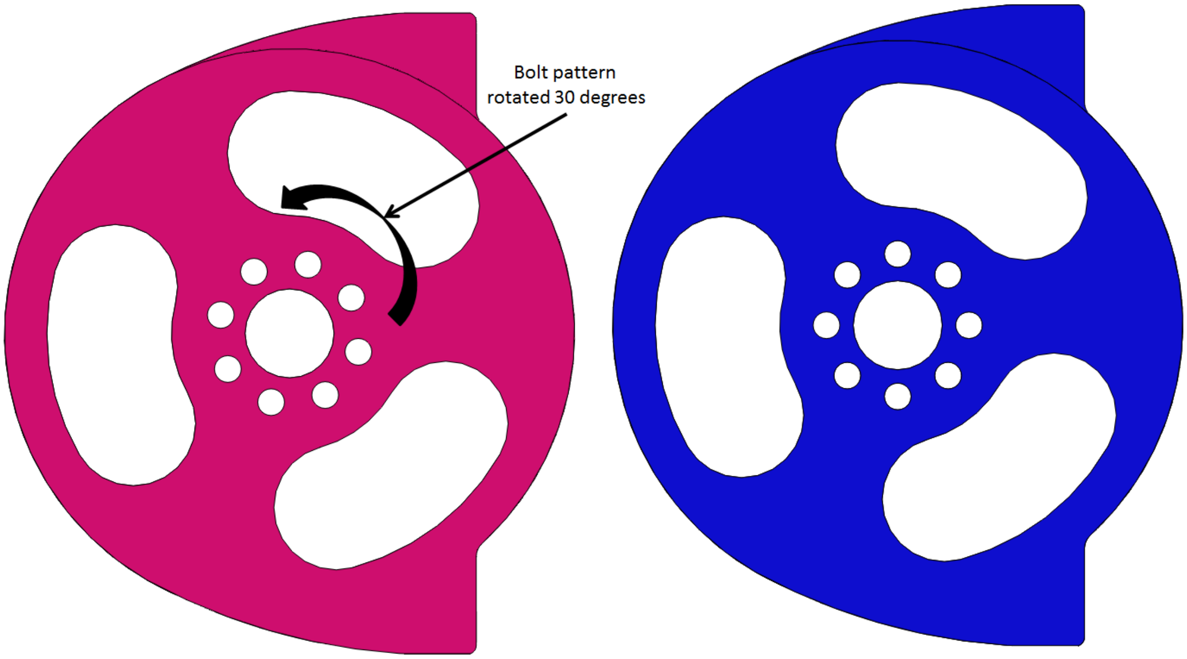


Figure 68 - Altered Bolt Pattern

Finally, the design of the motor allowed the pulley to recessed into the gearbox such that the overall profile of the motor – gearbox – pulley assembly was smaller. Additionally, since the motor pulley did not need to be pinned to anything, the design allowed for heavier pocketing which would reduce the overall weight of the pulleys. It should be noted, however, that an altered bolt pattern was created for the purpose of simplifying the machining. Finally, the motor pulley also needed an altered center bore diameter and keyway to account for the 22mm motor shaft. Since the team did not own a 6mm broach, the decision was made to simply use a smaller endmill and make a keyway in the non-traditional method. A final picture depicting the motor pulley design can be seen in Figure 69.

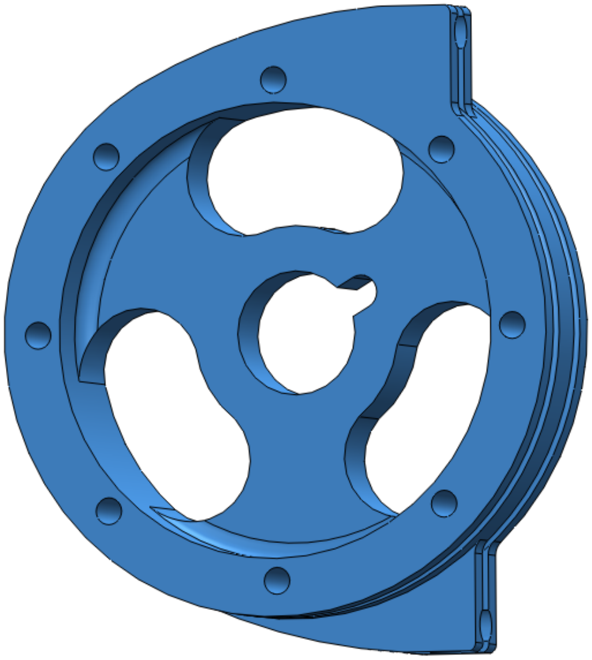


Figure 69 - Motor Pulley

The entire cable system was also prototyped in full scale to verify both the spring system and to check for interferences in such a system. Simple drafting tools can only lead so far and trying to attempt dynamic movement in these tools have proven to be exceedingly difficult. While the team did not model every cable used in the system into SolidWorks, construction planes were developed to check for interferences and general location of cable direction. The final full scale prototype was powered using 2 FIRST Robotics Globe Motors controlled with velocity control using a VEX system. A picture of the final full scale prototype developed out of MDF, acrylic and nylon string can be seen in Figure 70.

Criteria	Criteria Met?
Cables do no interfere	X
Springs compress when leg contacts ground	X
Tensioners have clearance and do no interfere with other components	X
Joint motions are independent	X

Table 15- Prototype Criteria

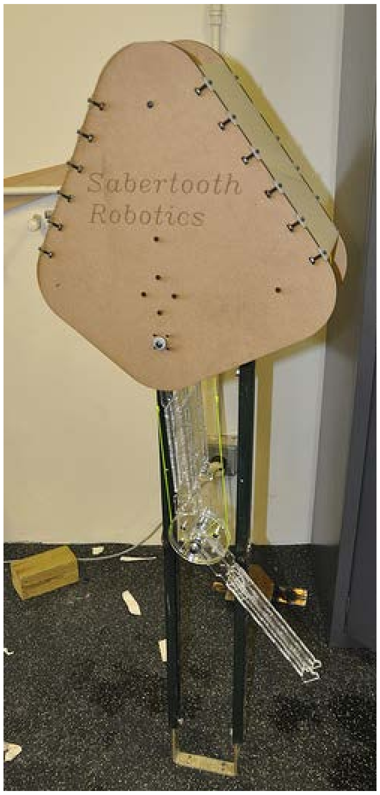


Figure 70 - Full Scale Prototype of Cable System

A number of criteria were tested with the prototype leg as shown in Table 15.

3.3.7 Ankle Design

Through background research the performance characteristics of the tendons and the ankle requiremetns realized there was an ability for a larger understanding of what would be effective on each ankle system. Ideally both the soleous and gastrocnemius would be recreated in both the front and rear ankle systems, unfortunately do to sizing requirments this option was too complex for this system. In order to decrease the complexity of the system but still provide that level of power output from the tendons it was decided that one tendon system could be implemented on each ankle system. This allowed for the rear ankle system to recreate the biarticulate muscle of the Gastrocnemius and the front ankle system to recreate the soleus for power generation. Each of these tendons were chosed based on their ability to statisfy the requirements listed in Table 7.

Once the correct muscle was chosen for each ankle system it was necessary to properly specificy the spring force that would be requried for this robot aplication. These force calculations were accomplished through the use of torque equations and by added multiplication factors in order to account for the special characterists of the front and rear legs.

The first tendon power which needed to be calcualted was the soleous tendon that was located on the front ankle system. The metatarsal lengths were used to construct a preliminary design. This design seen in Figure 71 used an extension spring located on the back of the front leg which would provide spring force when an upward force is applied to the base of the foot via the normal force of the ground acting on the robot.

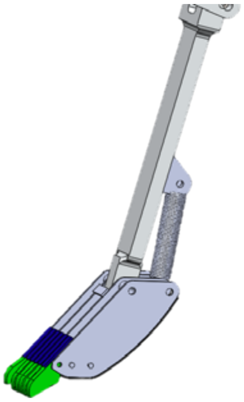


Figure 71 - Front Ankle V1

The force of the spring was calculated by applying a force to the moment arm that is created by the metatarsal plates and the lower leg segment. This moment arm can be seen in Figure 72, the dimensions correspond to the linear distance between the pivot and the external forces.

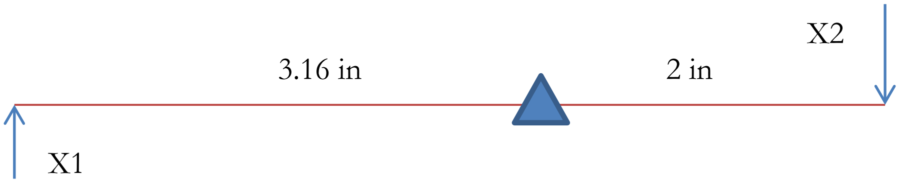


Figure 72 - Front ankle moment arm

The X1 force corresponds to the weight of a third of the robot. A third was selected to correspond to the fact that through a walking gait at least three legs are always in contact with the ground. The counteractive spring force corresponds to the X2 force located across the pivot of the foot.

$$90\ lb * (3.16\ in\ ) = X2 * (2.00\ in)$$

$$X2 = 126\ lbs$$

With the total spring force calculated it was also necessary to calculate the amount of travel the front foot was gain. The front foot has the ability to rotate about the end of the lower leg to aid in both mechanical compliance and power regeneration. This motion is dampened by the above calculated force, but the length of this motion still needs to be calculated. In order to maintain the proper length for the front lower leg a maxium motion of 35 degrees was allowed on the front foot. Using this angle of rotation a linear displacement of 1.25in was calculated for the front foot dampener.

Continueing with the desing process delivered a fairly substancial issue with decision to use an extension spring in the front ankle system. The first issue was discovered when discussing the system with Professor Robert Norton. Professor Norton made the team aware that the effective cycle span of a normal extension spring was very low. In addition when procuring the extension spring, it was discovered that any extension spring which would fall within the specifications for force and extension would not fit within the system. These two design resitrictions lead to exploring other dampending solutions, of these dampening solutions the two which would work the best were compression springs or gas springs. Unfortunately, both of these solutions required the response load on the feet to be move from one side of the leg pivot to the other. The foot plates were redesigned to allow use of gas springs as the dampening solution. The final design of the front ankle system can be seen in Figure 73. The reasoning behind the selection of the gas spring over a compression spring was because of the ability to swap the gas spring for a more powerful dampening force without any mechanical modification to the system.

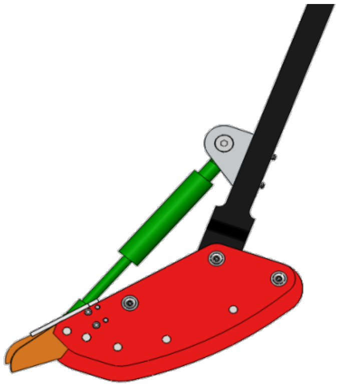


Figure 73 - Front Ankle V2

The second tendon power which needed to be calcualte was the gastrocneous tendon that was located on the rear ankle system. The use of a tendon modeled after the gastrocneous added a level of complexity to the basic design of the rear ankle system. This complexity is introduced because of the biarticulated nature which requires a connection directly from the ankle to the upper leg. This linkage fully controls the lower anatomical features of the rear leg and as was previously stated even a slight error with this ratio could highly reduce effeciency of gaits.

This biarticulate nature of the rear linkage needs to be carefully examined and with this examination a ratio needs to be established. This ratio corrisponds to angle between the uppler link and the lower link of the leg and the angle between the ankle and the lower link. This ratio is imperative to ensure proper gait control and that the ankle will not interfere with ground on the idle section of the gait. As seen in Figure 74 the rear ankle linkage consists of the three leg links (upper, lower, ankle) and a control link which will help ensure the ratio x will remain constant.

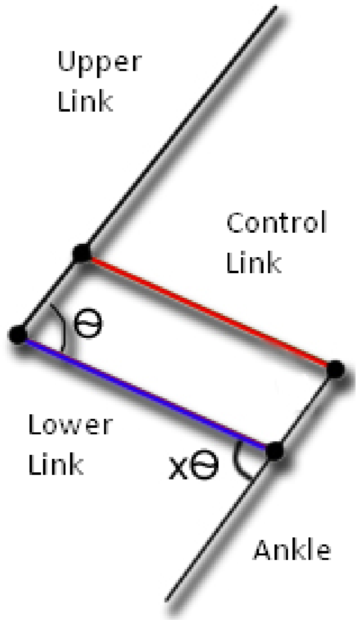


Figure 74 - Rear Ankle Linkage Model

The ratio between these two angles is a topic of dispute among academia. Unfortunately, this debate ensures that a perfect gait controller still is unable to be confirmed. One of the most reputable papers on gearing ratios within running dogs, *Dynamic Gearing in running dogs* (Gregersen, et al.,1998) explores that importance of power and efficiency within anatomical similarities. However, when discussing the ratio between the rear ankle and the upper link they even state that the ratio is 72%.(Gregersen, et al., 1998) Through further research it was found that this 72 was a self-referenced value, which was then referenced from an older study that has been referenced over 300 times.

This ratio was then used to design a full linkage for the rear ankle and this linkage can be seen in Figure 75. This linkage also employs a set of extension springs providing a spring force to the ankle in relationship to the upper link.

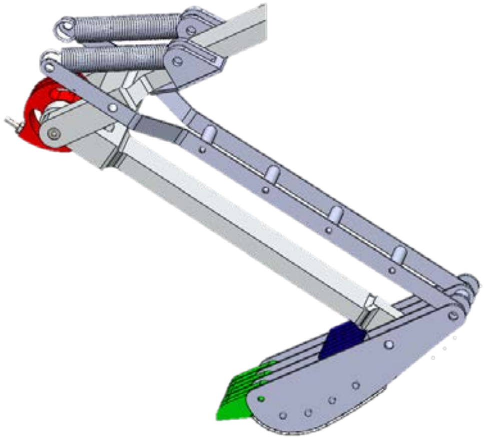


Figure 75 - Rear Ankle V1

Unfortunately after testing many gaits on the system it was becoming evident that the 72% ratio was incorrect. The rear ankle was not reaching the extension and tucking position that would be required to fulfill all of the gaits that the rest of the robot was designed to complete. In an effort to correct this improper ratio many videos of running dogs were examined. The videos had frames captured and each frame was compared to the new ratios of the ankle system that were being tested. Through some testing it was determined that a more realistic ratio to base a linkage ratio off of was determined to be between 81%. This new ratio was then implemented in the system and the rear ankle was complete.

As more analysis was performed on the rear ankle system, it became clear that the current spring setup was incorrect and would only influence the motor power consumption negatively. The Achilles robot provided a great model that could be examined in order to understand the biarticulate muscles. Through this model it was understood that placing a spring in series with the routing of the gastrocneous tendon would allow for the proper spring force to be created and the motor power consumption to be unaffected.

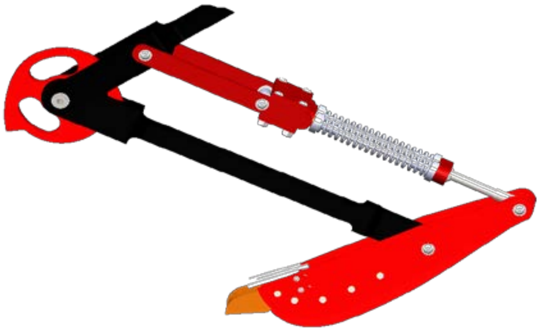
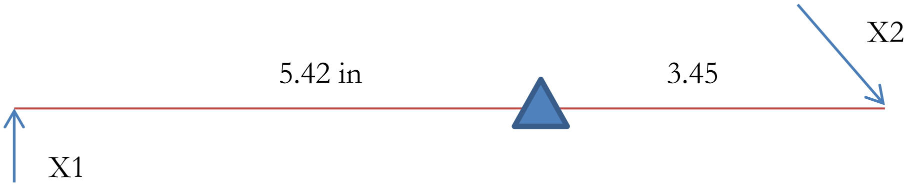


Figure 76 - Rear Ankle V2

The force of the spring was calculated by applying a force to the moment arm that is created by the metacarpal plates and the lower leg segment. This moment arm can be seen in Figure 72, the dimensions correspond to the linear distance between the pivot and the external forces.



The X1 force corresponds to the weight of a third of the robot. A third was selected to correspond to the fact that through a walking gait at least three legs are always in contact with the ground. The counteractive spring force corresponds to the X2 force located across the pivot of the foot.

$$90\ lb * (5.42\ in\ ) = X2 * (3.48\ in) * (\cos(60))$$

$$X2 = 230\ lbs$$

This total spring force will be acounted for in the final linkage system through the addition of two compression springs. These compression springs were specified based on two requirements, the first being the force and the second being the compression length. The rear foot has the ability to rotate about the end of the lower leg to aid in both mechanical compliance and power regeneration. This motion is dampened by the above calculated force, but the length of this motion still needs to be calculated. In order to maximize the power gained from the rear linkage a linear change in height was set at 4in for the end of the ankle. Using this angle of rotation a linear displacement of 2.5in was calculated for the rear foot dampener. This information allowed for Table 16 to be used to select LHC142J09M as the spring that will be used in the system.

P/N	Force	Compression
LHC187R07M	180	2.5
LHL1250A08	247	2.3
LHL1500A06	360	2.3
LHC142J09M	120	2.5

Table 16 - Rear Spring Selection

The curve which was generated previously corresponds to what would be the fluid motion of the foot regardless of the different bones it contains. This general form of motion contained within a curve allowed for the rear feet of the robot to be created in order to ensure fluid motion of the robots foot through its ground contact. The curve which was implemented from Figure 12Figure 12 on the robot can be seen in Figure 77.

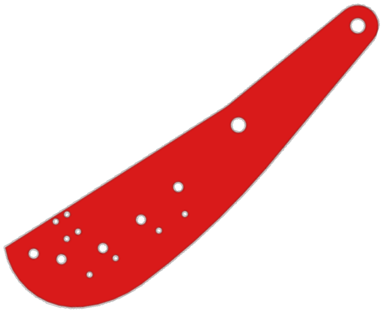


Figure 77 - Rear Foot Curve CAD

The final aspect of ankle design that must be accounted for is the legs ability to provide camber. The current foot design has a flat bottom which when the leg camber (Θ) increases will not provide adequate ground support. The leg camber (Θ) is able to produce values of both positive and negative 10 degrees as can be seen in Figure 78. This difference in camber requires foot ground interface to change in order to properly distribute the weight of the robot. This weight distribution is solved by adding a curve to the bottom of the feet. A curve matching the maximum camber in both directions ensures that the feet will always be in full contact with the ground plane.

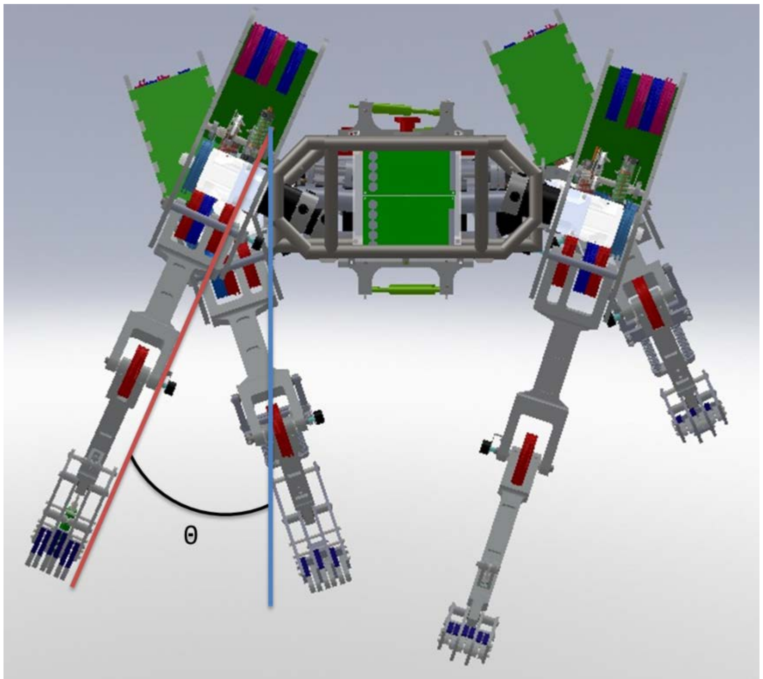


Figure 78 - Leg Camber Extremes

3.3.8 Manufacturing

Upon completion of the CAD model we recognized a need to make a strong manufacturing plan to quickly and accurately assess how to manufacture each part. This system proves to be crucial to make sure that parts could be manufactured in a quick manner while still achieving the required tolerances. The team identified every part to belong to at least one of the following categories: Laser-cutter part, Waterjet part, Mill Part, Lathe Part. Lasercut parts were generally identified as plain 2D parts to be made out of acrylic or ABS plastic. Waterjet parts, were constantly identified as parts which were also 2D but made out of either aluminum or steel. Lathe parts were identified as parts which has symmetry about an axis of rotation. Finally, mill parts were identified as the most complex parts which had complex features on any plane. Many parts also overlapped into more than one field with primary operations being completed on a lathe and then finish machining required on a mill. Once identified the part was then concretely analyzed to identify each step of the manufacturing, the required tolerance, as well as any custom fixturing which would be required to machine the parts. A strict manufacturing schedule spanning from January to March was developed to ensure that parts would be machined in a timely fashion. Roughly 150 unique (500 total) parts were manufactured in house, undergoing an average of 2 setups and totaling roughly 1500 total machining hours.

3.4 Electrical Integration

3.4.1 Saber Board

In an effort to properly interface with the National Instruments SBRIO both mechanically and through software, the team developed a PCB. The PCB was designed mainly as a breakout board, but also incorporated operational amplifiers for sensor inputs. The PCB design was developed in the EAGLE Layout Editor utilizing two of the 50pin connectors on the SBRIO. One of these connectors was primary used for Analog Inputs and the other was used for Digital I/O. In addition the board was designed to have a 5v input rail in order to provide the voltage necessary to run the many sensors attached to it. The final schematic for the Saber Board can be seen in Figure 79.

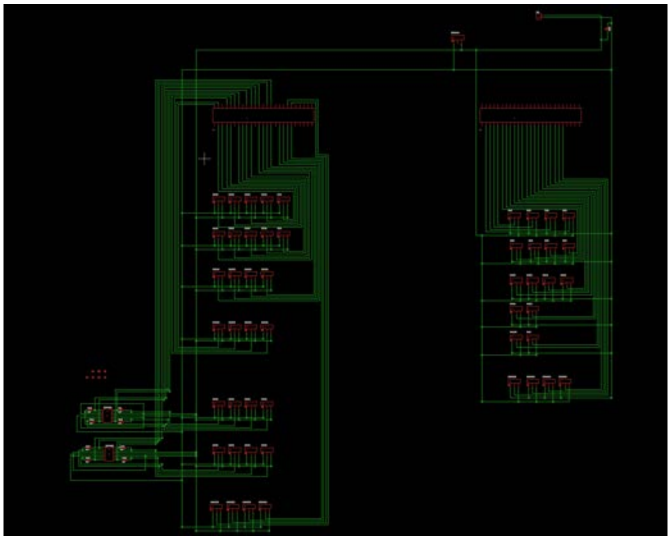


Figure 79 - PCB Schematic

With the schematic complete, it was necessary to create the footprints for the connectors which would be used. We used a female 50 pos ribbon cable to interface with the 50 pos connector on the SBRIO and JL molex connectors to provide three pin connectors. With these connectors the board layout was created and the programs auto routing feature was taken advantage of. While the auto routing feature was not perfect it did save some time. Two of the most important things that were done to the auto routing were to eliminate all sharp angles and to create both a ground and power plane on the top and bottom of board respectively. These planes also served to help reduce both high and low frequency noise on the system. The final board layout that was compiled can be seen in Figure 80, the red lines represent traces that were placed on the top and the blue lines are the traces on the bottom.

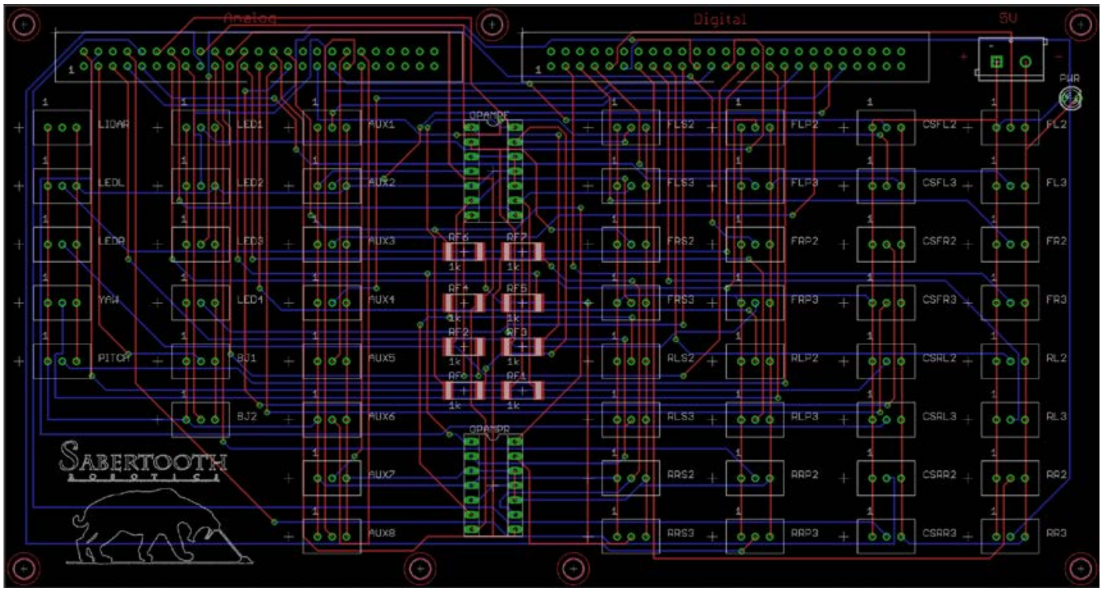


Figure 80 - PCB Layout

It is important to note the circuitry that was added to properly handle the varying resistors that would be used to measure the spring force within the spring pods. The varying resistor needed to be amplified in order to be properly read by the SBRIO ADC, this was accomplish through the use of the MCP 6004 operational amplifier. The boards were printed, assembled and tested as seen in Figure 81.

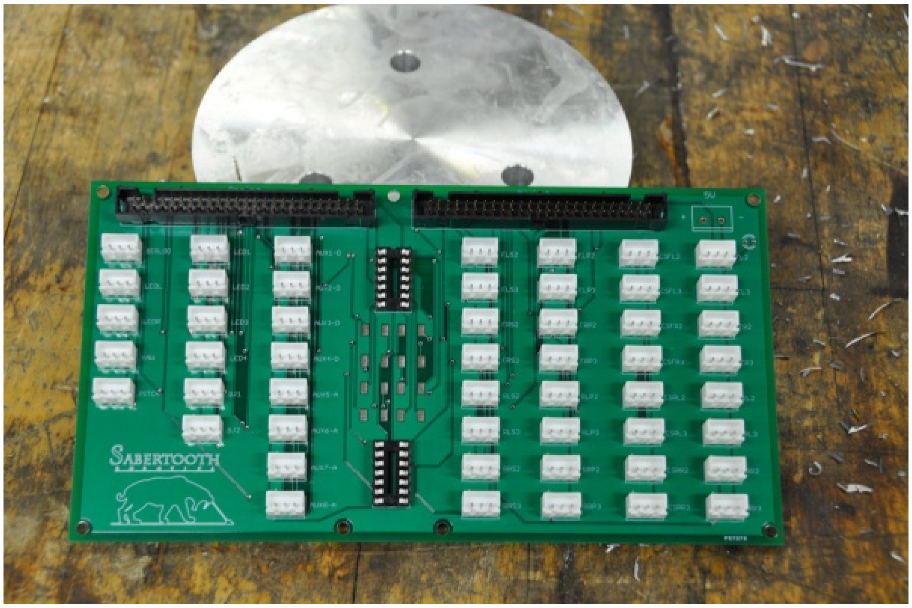
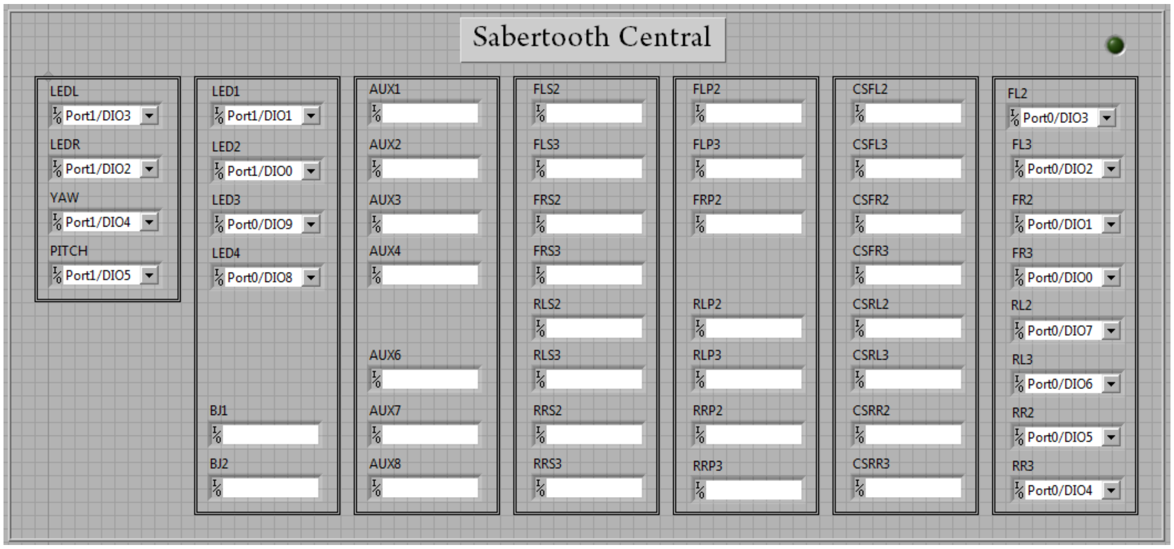


Figure 81 - Final PCB

While we were working on this PCB externally to the SBRIO, there was also a development plan for the LabView code which would interface with the PCB. The VI for this interface was setup to mimic the physical appearance of the PCB with the same naming scheme to ease the overall interface between the PCB and SBRIO. This will allow future calls to the I/O pins to be intuitive and reduce the amount of errors. In order to properly configure the VI (Appendix G) was created which lists which connectors are linked to which ports on the SBRIOs FPGA. The VI can be seen in Figure 82 and as you can see has the same row/ column layout as the physical PCB.


Sabertooth Central

LEDL Port1/DIO3
LEDR Port1/DIO2
YAW Port1/DIO4
PITCH Port1/DIO5

LED1 Port1/DIO1
LED2 Port1/DIO0
LED3 Port0/DIO9
LED4 Port0/DIO8
BJ1
BJ2

AUX1
AUX2
AUX3
AUX4
AUX6
AUX7
AUX8

FLS2
FLS3
FRS2
FRS3
RLS2
RLS3
RRS2
RRS3

FLP2
FLP3
FRP2
RLP2
RLP3
RRP2
RRP3

CSFL2
CSFL3
CSFR2
CSFR3
CSRL2
CSRL3
CSRR2
CSRR3

FL2 Port0/DIO3
FL3 Port0/DIO2
FR2 Port0/DIO1
FR3 Port0/DIO0
RL2 Port0/DIO7
RL3 Port0/DIO6
RR2 Port0/DIO5
RR3 Port0/DIO4

Figure 82 - PCB LABview VI

3.4.2 Electrical Architecture

The design of the electrical architecture of the robot was very important to ensure all of the sensors, controllers, and motors would be receiving the correct power and the signals would all function properly. This was very important for many reasons including physical requirements, amount of signals, ground loops and power interference. Once the mechanical systems were laid out it was evident that there were a large amount of sensors that would be required and the motors would require unique motor controllers.

The initial control platforms that the electrical system began with were a NI SBRIO 9632xt and an Asus EEE pc. In addition motor controllers for the Maxon motors, motor controllers for the Ampflow motors, sensor input and servo output were interfaced with the central control system. The total number of sensors and controllers that require power in the system can be seen in Table 17.

Sensors & Controllers	Quantity
Joint Potentiometer	8
Varying Resistor	8
Body Joint Potentiometer	2
Motor Current Sensor	8
Limit Switch	4
Maxon Controller	8
RoboteQ Controller	8
LED Controls	4
Servo Control	2
LIDAR Power	1
SBRIO 9632xt	1

Table 17 - Electrical Requirements

It is critical to note that many of these systems have different power requirements and that a dc dc converter would be necessary to properly power the system. It was calculated that the 1A onboard 5v power from the SBRIO was not enough current to drive all of the sensors. It was decided that 5A was required to properly power the 5v systems on the robot. This requirement matched the specs of the current logic step-down converter in Figure 83.

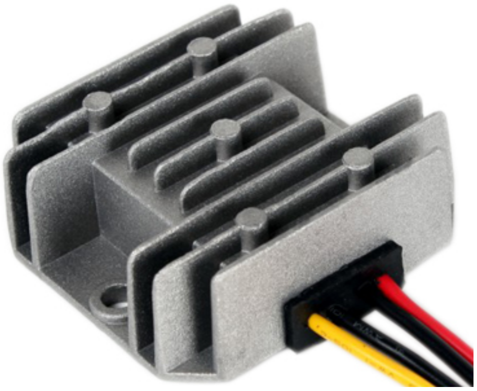


Figure 83 - 28v - 5v DC DC

With the power requirements taken care of, it was necessary to find a solution to the large number of signals that were required to make the robot function. This provided for two unique problems that needed to be solved, the first being organization and the second being noise in the system. All of the signals being input or output needed to be run through the SBRIO and therefore the organization was an easy solution. In order to organize the inputs to the 50 pos connectors on the SBRIO a custom Saber board PCB was created to handle all of the sensor input and output in an organized fashion.

With a fully organized signal management system in place, it was necessary to look into the power distribution. This system contains sensors and controllers with very different power requirements both in terms of voltage and current supply. These power requirements made it necessary to divide the power system into three distinct systems; a 28 volt un-fused system, a 28 volt fused system and a 5 volt system. The power and signal wiring diagram can be seen in Figure 84.

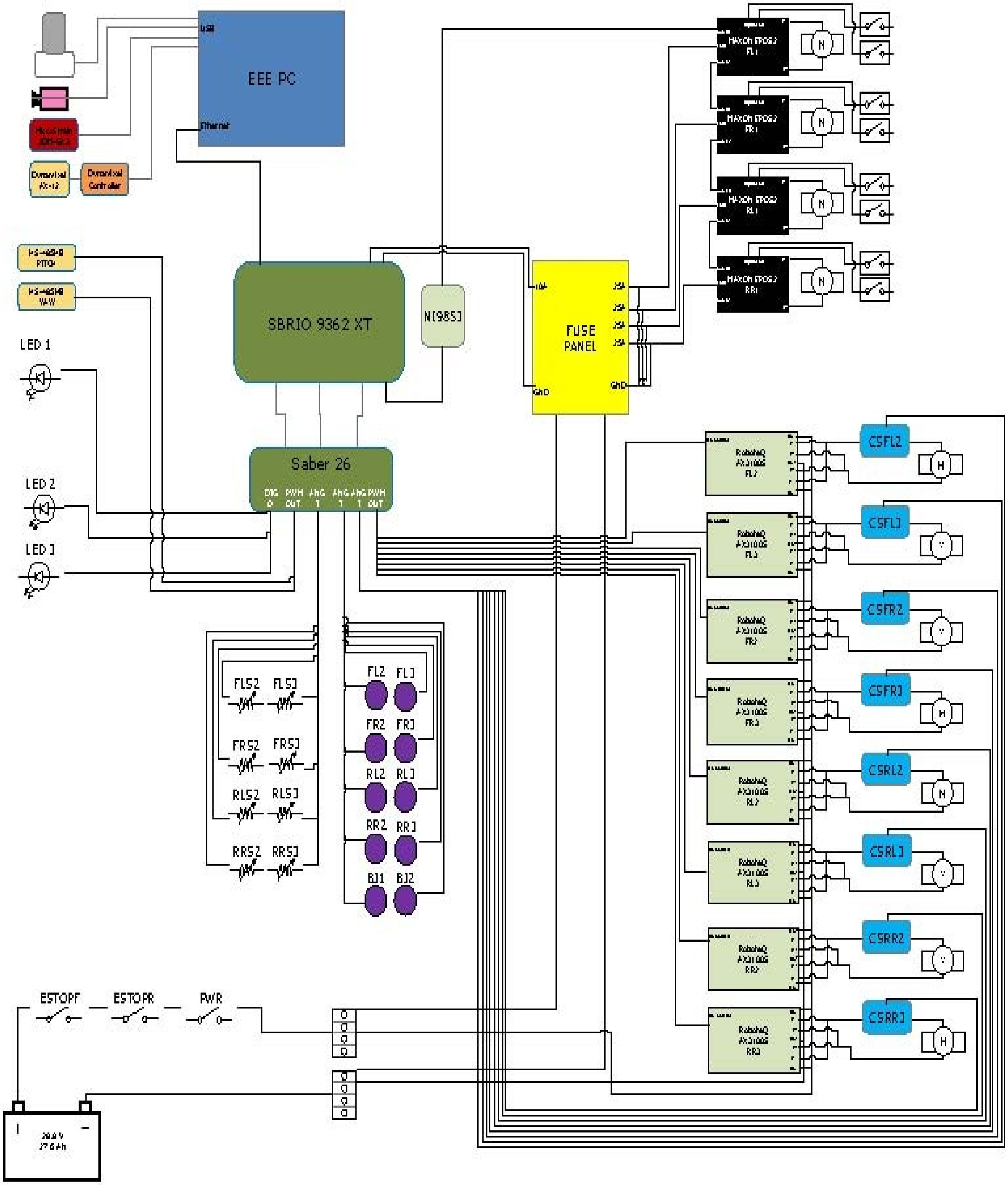


Figure 84 - Wiring Diagram

Taking a closer look at this wiring diagram we can see the full layout of sensors and controllers. Starting with the Asus EEE pc the robot connects to the Hokuyo LIDAR, PS3 eye camera, IMU and Dynamixel servo which controls the tilt of the LIDAR. Of these sensors it is necessary to understand the unique power requirements of the LIDAR and the dynamixel servo. Both of these systems require a voltage of 12v and this power was supplied by a separate power supply in the form of a 2100mah Lithium Polymer battery located within the sensor head. The EEE pc operators off of its own separate battery and needs not be connected to the robots onboard power supply.

The EEE pc then connects to the SBRIO 9632 which is the bases for the rest of the system. The SBRIO is connected to the 28v fused power bus and is protected behind a 10amp fuse. In addition the SBRIO commands three signal lines. The digital and analog signal lines are connected to a custom printed Saber board which organizes the systems input and output to controllers and sensors. The third signal line connects to a NI 9853 expansion module. This module is in charge of maintaining and openCAN bus to our onboard Maxon controllers. The module is powered directly off of the SBRIO and connects directly to an expansion serial port.

Power requirements for the rest of the system depend highly on the connections to the Saber board. The first power consumption comes from the robots status LEDs and headlights. These LEDs are used to maintain a constant status light as well as light the front of the robot for the camera's clarify. All of these LEDs are connected to the fused 28v line through a 28v relay which is controlled via the digital outputs of the SBRIO/ Saber board. The status LEDs change color corresponding to Table 18.

Status	LED Action
Standby	Blinking Green
Running	Steady Green
Error	Blinking Red
Emergency Stop	Steady Red

Table 18 - LED Status

The next signal and power system came with the various sensors which were required for robot operation. These sensors including potentiometers, current sensors, varying resistors and limit switches all required a 5v source as well as a signal connection to the SBRIO. However, all of these sensors would prefer to have both the power and signal to come from the same centralized location in order to keep the wires together. This was accomplished through the combination of power and signal in the Saber board. The Saber board provided the signal connection to the SBRIO as well as a common 5V power plane connection to the 28v to 5v dc-dc converter.

The Maxon motors driving the custom linear actuators for the hips needed to be controlled with the Maxon controllers. These controllers have specific power and signal requirements. The controllers are capable of being daisy chained for controlled if connected to an openCAN bus. This is accomplished through the use of a NI 9853 module in conjunction with the SBRIO 9632. The Maxon controllers power requirements ensure that the controllers and motors would not suffer overdraw damage. This required the controllers to be connected to the 28v fused bus behind 25 amp fuses for each controller.

The Ampflow motors driving the shoulder and knee joint needed motor controllers which would be capable of the current spikes that may be seen during extreme gaits. It was calculated that the motor torques required to go through the gait would have an average power consumption of 50amps with a peak around 80a. This was calculated based on Equation 7.

$$\text{Current Draw}Current = \frac{1.9\ hp}{28\ v} = 50\ amps$$

Equation 7

In order to account for these current draw Roboteq 3100s controllers were chosen. These controllers are controlled via a simple PWM signal with a 2ms frequency. This signal was generated by the SBRIO and passed through the Saber board to each of the controllers. With the high current requirements of these controllers it was necessary to look into attaching them to the un-fused 28v line. One unique feature of this controller was internal current limiting and because of the controller's ability to limit the draw on the motors to below 120amps, we decided that it was unnecessary to have the motor controllers fused.

The next problem that needed to be handled was that of noise within the system. The largest cause of noise in this system was the high power and current that existed within the main power lines of the system. The amount of current being run through the power and ground lines needed to be isolated from the signal lines because of the low voltage that they were providing to the analog to digital converters on the SBRIO. This was accomplished by carefully laying out the wiring for the robot. As can be seen in Figure 85 the power and ground was run on the top of the robot while the signal wires (input/output) were run along the bottom of the robot.

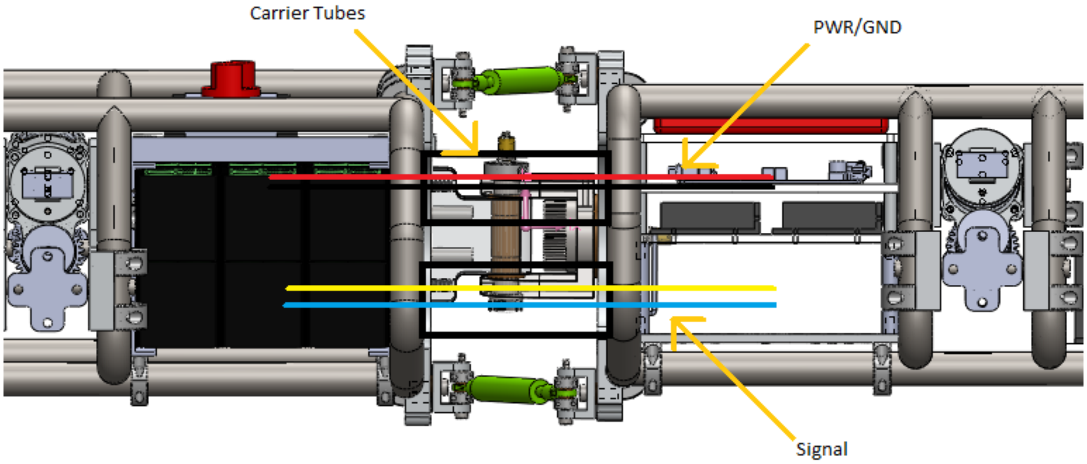


Figure 85 - Power/ Signal Layout

In addition to this isolation method the signal wires were all 4 core twisted pair shielded wires. We decided that running 22 AWG braided wire would be the most flexible wire for this robot and would easily be capable of handling the current that would be seen.

With both the control and power systems organized in the robot, it was necessary to add in some safety features. These safety features came through the introduction of emergency stop switches to the system. We decided that because of the size of this robot there needed to be a total of two emergency stops on the robot, one being located at the front of the robot with the other being located at the back. These emergency stop locations would enable a person to reach an emergency stop regardless of how the robot is approaching that person.

Because of the high current requirements of the robot, the emergency stops were required to be connected to a relay bank in order to properly stop motion of the motors. The emergency stop when depressed would stop all power going to the motor from the motor controller. This is accomplished through wiring a relay inline between each motor controller and motor. This series of relays would allow the low current capability of the emergency stop to handle the high current capability of the robot.

3.4.3 Motor Controllers

An H-bridge is the simplest form of motor control which works on the concept of alternating gate control and therefore electrical flow through the motor. The H-bridge circuit seen in Figure 86 is an example of a very basic H-bridge control circuit.

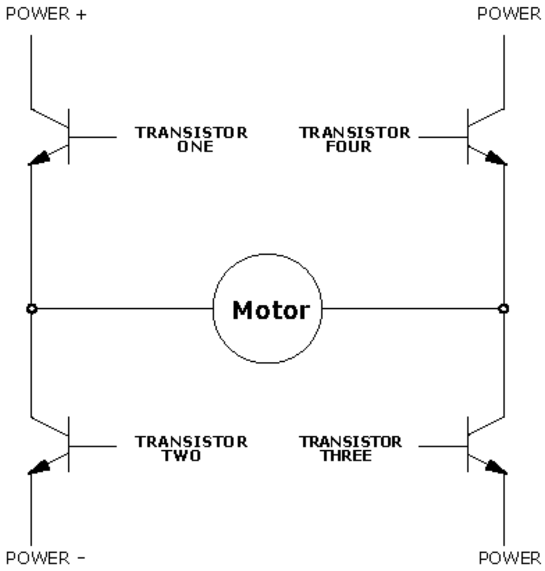


Figure 86 - H-bridge

By opening and closing transistor one/transistor four and transistor two/ transistor three respectively a steady motor control can be established. This control is established through a PWM signal which is a square waveform with a given duty cycle. The duty cycle is used to control both the speed and direction of the motor.

With more complex motors in the robot this control system needs to have added elements in order to ensure proper smooth motor output. The robot used two different types of motors for actuation. The movement of the hips linear actuators was accomplished through the use of Maxon ECMax40, which are brushless DC motors with shaft encoders. The movement of the remaining shoulders and hips is accomplished through the use of Ampflow A28-150, which are brushed DC motors.

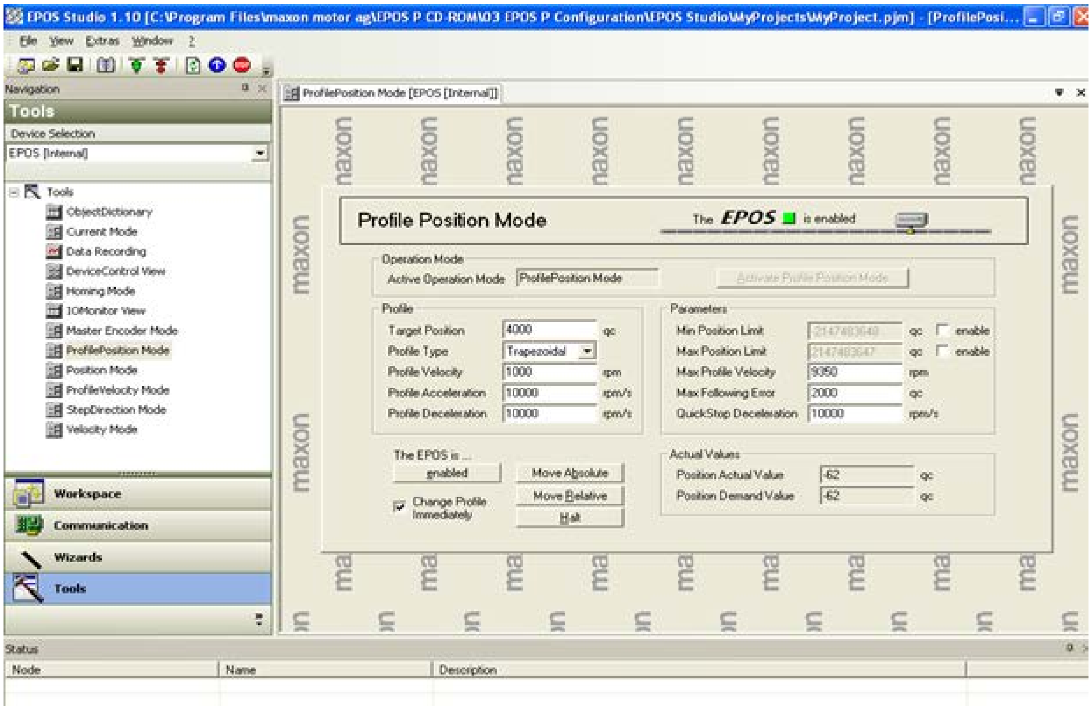
Brushless and brushed motors require different control systems on their respective motor controllers. These facts lead to the careful selection of the motor controllers for each type of motor on the robot. The first motor in which the control system was investigated was the Maxon ECMax. This motor was more restricted in its controller selection because of its brushless design. The first controller that was explored was the Maxon EPOS2 70/10, which was the controller that was the Maxon controller properly paired with the ECMax and can be seen in Figure 87.

Figure 87 - Maxon EPOS2 70/10

This controller had all of the required connectors to provide proper velocity, position and acceleration control of the ECMax. This feature was something that could not be matched with any other comparable motor controller. With these built in features it was decided that the time saved would be worth the additional cost of the controllers. Fortunately these controllers were available to the team for highly reduced cost and could be accounted for in the budget.

These motor controllers provided a unique control system which allowed for all four controllers to be linked on the same communication line. This is made possible via the controllers ability to communicate on an open CAN bus. These controllers are setup in a master slave configuration in which the first controller acts as the initial master link to the rest of the controllers. These controllers are then daisy chained together, that being each is connected to the controller previous and next in line. The final controller has the DIP switches toggled to add a resistance to the CAN bus which indicates a final node to the system.

This openCAN system was tested through the use of the EPOS studio software seen in Figure 88. The software allowed for the EPOS2 controller to exhibit both position and velocity control, in conjunction with outputting the encoder outputs of the motor. With the initial tests complete the software was then used to test the full openCAN system. Multiple nodes were setup through the software and multiple motors were capable of being commanded at different velocities to differing positions simultaneously.


EPOS Studio 1.10 [C:\Program Files\maxon motor ag\EPOS P CD-ROM\03 EPOS P Configuration\EPOS Studio\MyProjects\MyProject.pjm] - [ProfilePosi...

Profile Position Mode

The EPOS is enabled

Operation Mode

Active Operation Mode: ProfilePosition Mode

Profile

Target Position	4000	qc
Profile Type	Trapezoidal	
Profile Velocity	1000	rpm
Profile Acceleration	10000	rpm/s
Profile Deceleration	10000	rpm/s

Parameters

Min Position Limit	-2147483648	qc	□ enable
Max Position Limit	2147483647	qc	□ enable
Max Profile Velocity	9350	rpm	
Max Following Error	2000	qc	
QuickStop Deceleration	10000	rpm/s	

The EPOS is ... enabled

☑ Change Profile Immediately

Actual Values

Position Actual Value	-62	qc
Position Demand Value	-62	qc

Figure 88 - EPOS Studio

The second motor in the control system was the Ampflow A28-150. This motor is very easy to control with a simple motor control because of its brushed design. However, the restrictive properly of this motor is the amps that it will be drawing on a constant and pulse rate. The system design requires a motor controller that can provide 50A draw continuous and peaks above 70A.

This led to a rigorous selection process that brought together motor controllers such as Sabertooth 2x50, Robot Power Sidewinder, Ampflow Dual Motor Controller and Roboteq 3100s.

The first motor controller which was explored was the Sabertooth dual 50A motor driver. This motor controller is able to supply two DC brushed motors with 50A continuous with the ability to peak up to 100A per channel for a few seconds. There is also built in overcurrent and thermal protection which will work to protect the controller in the event of a crash. The controller is capable of communications via analog voltage, radio control, serial and packetized serial.(DimensionEngineering, 2007) This controller seen in Figure 89 was originally chosen by the team, but stocking restrictions were unable to be used as a final solution.

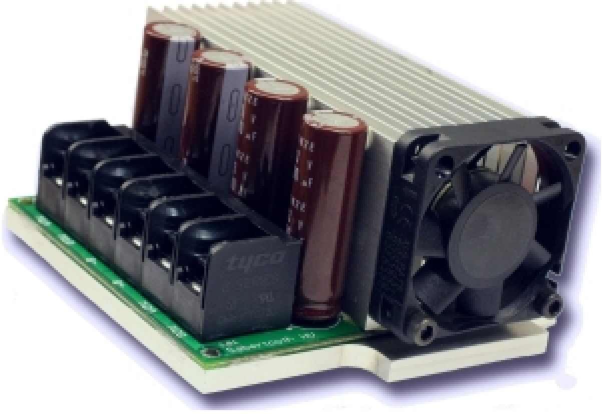


Figure 89 - Sabertooth 2x50

The second motor controller which was investigated was the Robot Power sidewinder. This motor controller is able to supply two DC brushed motors with 80A continuous per channel with the ability to peak up to 150A for five seconds. There is a built in current limiting ability adjustable from 10A to 130A as well as thermal protection to protect the controller in the event of a crash. The controller is capable of communications via R/C pulse standard(PWM), and TTL serial. (Robot Power, 2003) This controller has a built in failsafe that shuts off the motor if R/C signal is lost. This controller seen in Figure 90 was chosen as an alternate choice to the Sabertooth controller.

Unfortunately the sidewinder controller was discontinued and was unable to be used as a final solution.

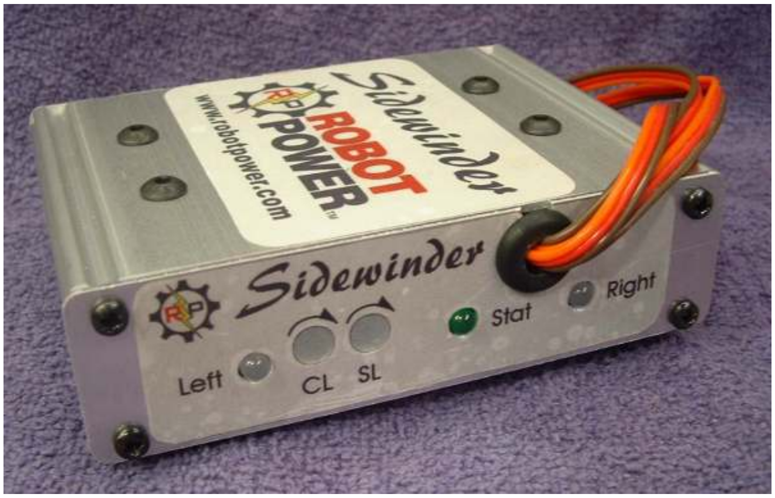


Figure 90 - Robot Power Sidewinder

The third motor controller which was investigated was the AmpFlow dual motor controller. This motor controller is able to supply two DC brushed motors with 80 Amp continuous per channel with the ability to peak up to 120A for a few seconds. There is built in over-current and over heat control logic which prevents meltdowns and provides the maximum current even under stalled conditions. This controller is capable of control via PWM, torque, velocity and servo-position. These control loops are command via RS232, RC or PWM input (AmpFlow, 2005). This controller seen in Figure 91 was chosen as an alternate choice to the Robot Power Sidewinder. Unfortunately the AmpFlow controller was out of stock indefinitely and was unable to be used as a final solution.

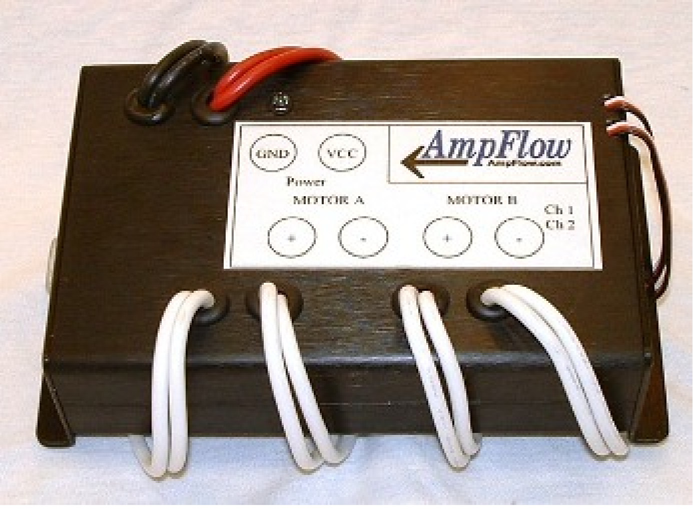


Figure 91 - Ampflow Dual Motor Controller

The fourth motor controller which was investigated was the RoboteQ single channel motor controller. This motor controller is able to supply one DC brushed motor with 120A continuous with the ability to peak at 500A for a few seconds. These controllers have the ability for full MCU digital design that being the ability to have fully programmable operation of advanced algorithms. In addition the controller is capable of open loop/closed loop speed control and closed loop position control. The controller is capable of communications via analog voltage, radio control, serial, wireless modem and packetized serial. The onboard software is capable of field upgradable software ensuring that features are always up to date.(RoboteQ, 2005) The controller seen in Figure 92 was chosen as an alternate to the AmpFlow dual motor controller. Fortunately these motor controllers were available and therefore were used as a final control system.

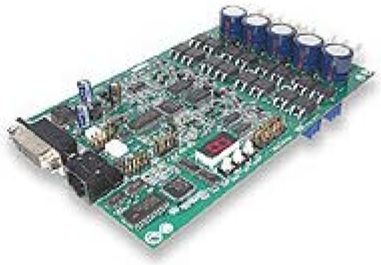


Figure 92 - Roboteq 3100s

With the selection of the RoboteQ 3100s it was necessary to figure out the proper way to send control signals in order to reduce the amount of lost time in the system. The first process that needed to be understood was controlling one controller via PWM. This was accomplished by initially correctly connecting the RoboteQ controller to an RC input. However before using the pinout seen in Figure 93 it was necessary to understand how the channels on these motor controllers operated.

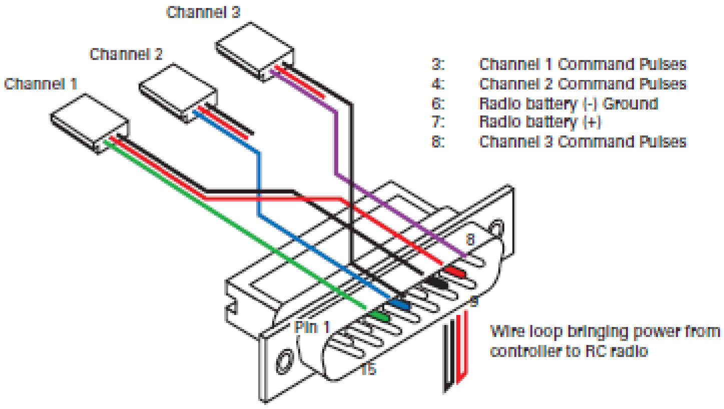
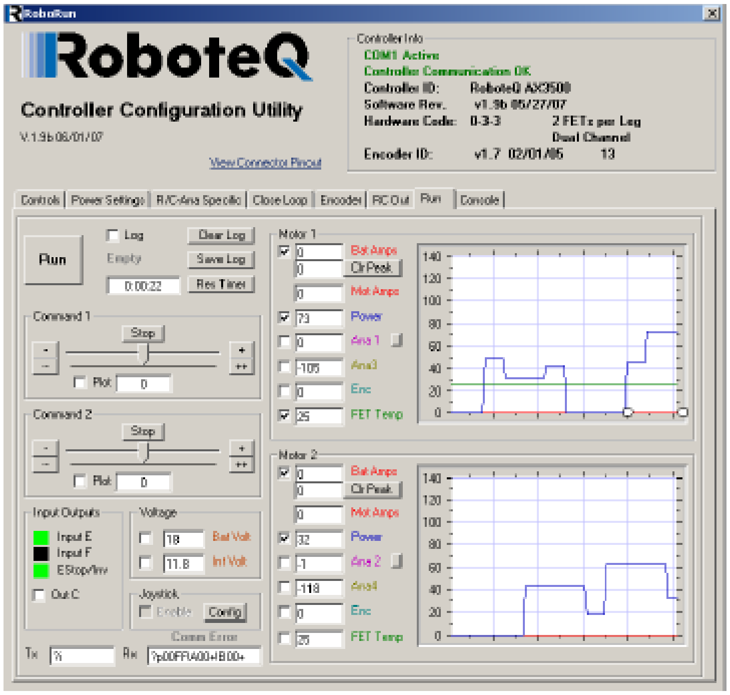

Channel 3
Channel 2
Channel 1
3: Channel 1 Command Pulses
4: Channel 2 Command Pulses
6: Radio battery (-) Ground
7: Radio battery (+)
8: Channel 3 Command Pulses
8
9
Pin 1
15
Wire loop bringing power from controller to RC radio

Figure 93 - 3100s Pinout

The RoboteQ 3100s is advertised as a 120A single channel controller, but also is sold as the RoboteQ3500 which is a 60A dual channel controller. It was determined that the motor needed to be connected to both channel outputs on the controller, but could be controlled simply thought one channel input to the controller. This was accomplished by jumping pin 8 and 9 on the controller to indicate single channel control mode.

With the controller setup for single channel output and wired to the output of the computer, it was necessary to test the firmware and output of the controller. This was accomplished through the use of the RoboteQ Controller Configuration Utility seen in Figure 94. The software allowed for all of the onboard capabilities to be tested and upgraded if necessary. In addition it allowed for the computer to control the motor controllers output and graph its change in power output over time.



V.1.9b 06/01/07

Controller Info
COM1 Active
Controller Communication OK
Controller ID: RoboteQ AX3500
Software Rev. v1.9b 05/27/07
Hardware Code: 0-3-3 2 FETs per Leg
Dual Channel
Encoder ID: v1.7 02/01/05 13

□ Log
Empty
0:00:22

Command 1
□ Plot 0

Command 2
□ Plot 0

Input Outputs
🟩 Input E
■ Input F
🟩 EStop/Inv
□ Out C

Voltage
□ 18 Bat Volt
□ 11.8 Int Volt

Joystick
□ Enable

Comm Error

Tx %
Rx ?p00FFIA00+IB00+

Motor 1
☑ 0 Bat Amps
0
0 Mot Amps
☑ 73 Power
□ 0 Ana 1
□ -105 Ana3
□ 0 Enc
☑ 25 FET Temp

Motor 2
☑ 0 Bat Amps
0
0 Mot Amps
☑ 32 Power
□ -1 Ana 2
□ -118 Ana4
□ 0 Enc
□ 25 FET Temp

Figure 94 - RoboteQ Configuration Utility

With the controller was successfully controlled by the RoboteQ software it was necessary to move to PWM control from the FPGA. The PWM output which was created by the SBRIOs FPGA was based off of the template seen in Figure 95. This template explores the full range of duty cycles that the controller would expect for control of the motors. This range varies from a waveform of 1ms corresponding to full reverse to a waveform of 2ms corresponding to full forward. These extremes are based around the neutral command of 1.50 ms or command value of 127.

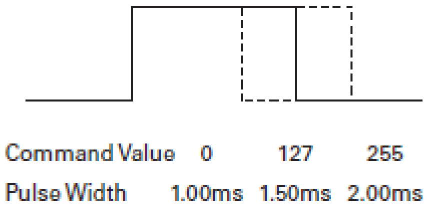


Figure 95 - RC Pulse Output waveform and timing

With successful control of a single 3100s it was necessary to design the final control architecture for all eight motor controllers. The RoboteQ controllers have the ability of controlling eight other RC driven devices from their own outputs. This ability was perfect to control all eight motor controllers with a single RC output from the SBRIO FGPA. Unfortunately after researching exactly how to control this feature, it was found that when using a system such as this the refresh time of the control loop on the motor controller moved changed from 2ms to 18ms. This change in control loop time was due to the manner in which the PWM waveform was passed from the master controller to the slave controllers. This is illustrated in Figure 96 showing the initial control PWM to ch1 controller and then the subsequent control signals to the other seven channels. Each waveform cannot be sent until the previous waveform is finished being read by the previous controller. Utilizing a system like this places a full lag of 18ms into the control loop.

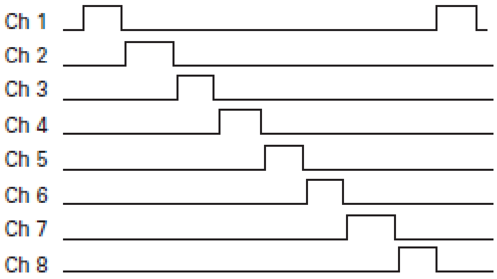


Figure 96 - Sequential Pulses Per Channel

In order to correct for the control loop lag created through the transition between master and slave controllers, it was necessary to switch the control logic of the controller fully to the SBRIO FPGA. Switching the control of the PWM for all eight controllers to the SBRIO FPGA allowed for the simultaneous control of each signal. This was accomplished by connecting each individual controller signal to a digital output on the SBRIO FGPA through the Saber Board. These individual digital outputs can each be set at the same time allowing for a full control loop to operate within a single 2ms time frame.

3.4.4 Power System

In order to meet the design requirements an onboard battery system would need to be selected for the purpose of powering the entire system. Knowing that the motors in a 5ft/sec walking gait would draw on average 50amps of power, the team would have to look into batteries with high amounts of energy density. The team explored traditional battery technologies including Sealed Lead Acid(SLA), Nickel Cadmium (NiCd), Nickel Metal Hydride (NiMh) and Lithium Ions. Quickly it was clear that exploring lithium technology would be the only viable option. Two specific lithium technologies were explored in great detail and found to be feasible for this project – specifically lithium polymer gel cell batteries (LiPo) and lithium iron phosphate batteries ($LiFePO_4$). The choice was quickly made to eliminate lithium polymer batteries due to their generically dangerous nature. Combining so many high power cells which are already volatile by nature could likely end in disaster (ThunderPower). As such, lithium iron phosphate batteries were further explored as the main power source. The team found that the most consistent and reliable company manufacturing such batteries was A123 Systems. Upon a few short conversations the team was able to form a partnership with A123 and work together with them to pick out the correct cells needed for the application. The team chose to go with 96 A123 26650 cells. These cells would offer the capabilities shown in Table 19. The final configuration chosen for the 96 cells was to put them in an 8S12P configuration where we would have 12 packs in parallel with one another each containing 8 cells in series.

Statistic	Single Cell	Full 8S12P Configuration
Capacity	2.3Ah	27.6
Nominal Discharge	69A	828A
Burst Discharge	138A	1656A
Nominal Voltage	3.3V	26.4V
Peak Voltage	3.3V	28.8V
Weight	70g	6720g

Table 19 - A123 Cell Capabilities

While the system did not incorporate any balancing capabilities on board, the team did proceed to buy advanced chargers specifically designed for rapid charge of the A123 battery cells. These chargers (iCharger 208B) would be able to charge the batteries at a 20A charge rate allowing for the battery to go from 0% life to 100% in roughly 10 minutes. More importantly, the chargers also allowed the batteries to have each cell actively balanced.

In addition to the 24V system, the team also had one 12 volt battery powering the LIDAR unit. While a 24V-12V DC convertor could be used, the team found it much easier to place a small lithium polymer battery right next to the LIDAR unit. While the lithium polymer battery still ran into the same risks of dangerous and possibly volatile chemistry the relative power draw of the LIDAR unit (1 Amp) was significantly lower than the overall potential power output of the battery (40 amps). As long as the team was careful to ensure that the battery did not drop down below the recommended 3.3V per cell then there would be no problems with overdraw which would lead to possible fire.

3.4.5 NI sbRIO 9632

The sbRIO 9632 shown in Figure 97 integrates a real-time processor, a field programmable array, and I/O on one board. Utilizing all three of these embedded systems to compute lower level functions of the robot ranging from motor control to data acquisition allows near instant reaction time on the robotic platform.

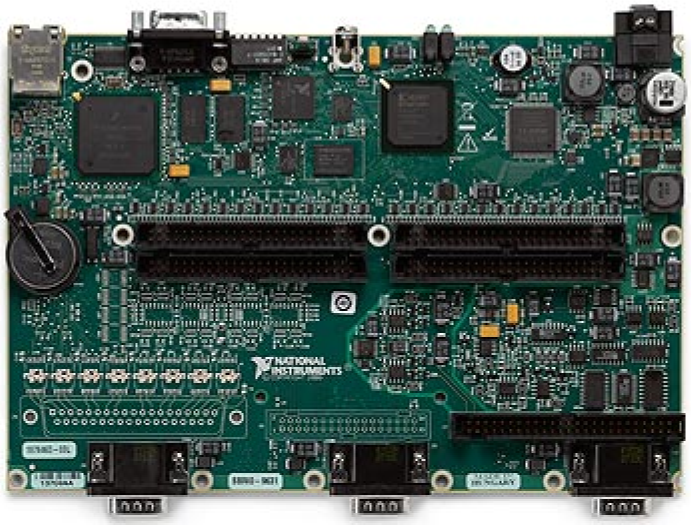


Figure 97: sbRIO 9632

- 400 MHz processor, 256 MB nonvolatile storage, 128 MB DRAM for deterministic control and analysis
- Integrated 2M gate reconfigurable I/O (RIO) FPGA for custom timing, inline processing, and control
- 110 3.3 V (TTL/5 V tolerant) DIO lines, 32 16-bit analog inputs, four 16-bit analog outputs
- 10/100BASE-T Ethernet port and RS232 serial port

The real-time processer is in charge of receiving packets of data from the netbook with motor positions every 50ms through UDP packets. Acting as a medium, the processor passes information from the computer to the FPGA. As this data streams to the processor, it computes how to handle this information based on several conditional statements regarding current, and position control.

The FPGA has 2M reconfigurable gaits. All of these gaits or integrated circuits are designed to be configured by the customer or designer after manufacturing. In our case, we have used these gaits and reconfigured them to perform very low level processing for 8 separate motors and data acquisition which gets sent up the chain to the netbook.

The I/O lines contain 110 3.3V digital I/O ports, 32 single-ended/16 differential 16-bit analog input channels, and 4 16-bit output channels. We used 26 analog inputs, 4 digital inputs, and 8 digital outputs. Inputs were used to acquire data from 16 potentiometers, 8 current sensors, and 8 limit switches(2 wired to one input). Outputs were used for PWM signals to Roboteq motor controllers, status LEDs, and Servo Control for the LIDAR and Camera.

3.4.6 NI 9853, CAN Module

The NI 9853 has 2 high-speed CAN ports capable of 1Mbps transfer speeds through a DB9 port

Figure 98: NI 9853 Can Module

As the real-time processor calculates the positions for the hips, it creates a CAN frame with position data which is passes to the EPOS2 Motor Controllers. This CAN frame is sent to the necessary node accepting the information.

3.4.7 LabVIEW Overview

LabVIEW is a graphical programming environment that is used to create a flowchart-like program. It uses G-code, expressed in G-units. LabVIEW is commonly utilized for data acquisition, instrument control, and industrial automation. LabVIEW programs are called Virtual Instruments, or VI for short. Each VI contains three components: a block diagram, a front panel, and a connector panel. Each VI can be implemented in other VI's with the connector panel allowing for simple scaling and embedded testing.

Sabertooth's application of LabVIEW required a large amount of low level processing that would be split into four layers: Networking(Figure 99), Motors(Figure 100), Interface(Figure 101), and Control(Figure 102).

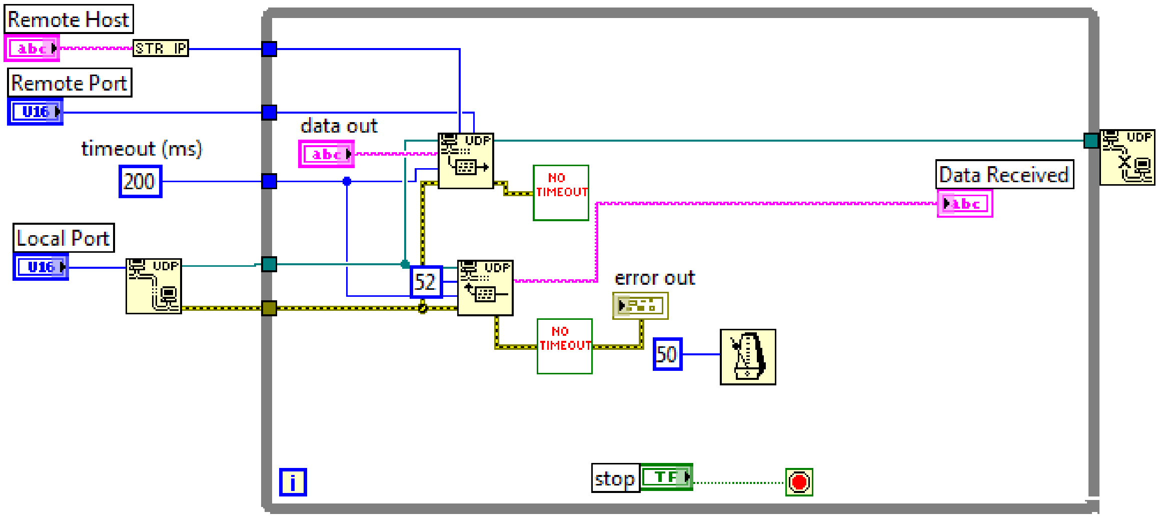
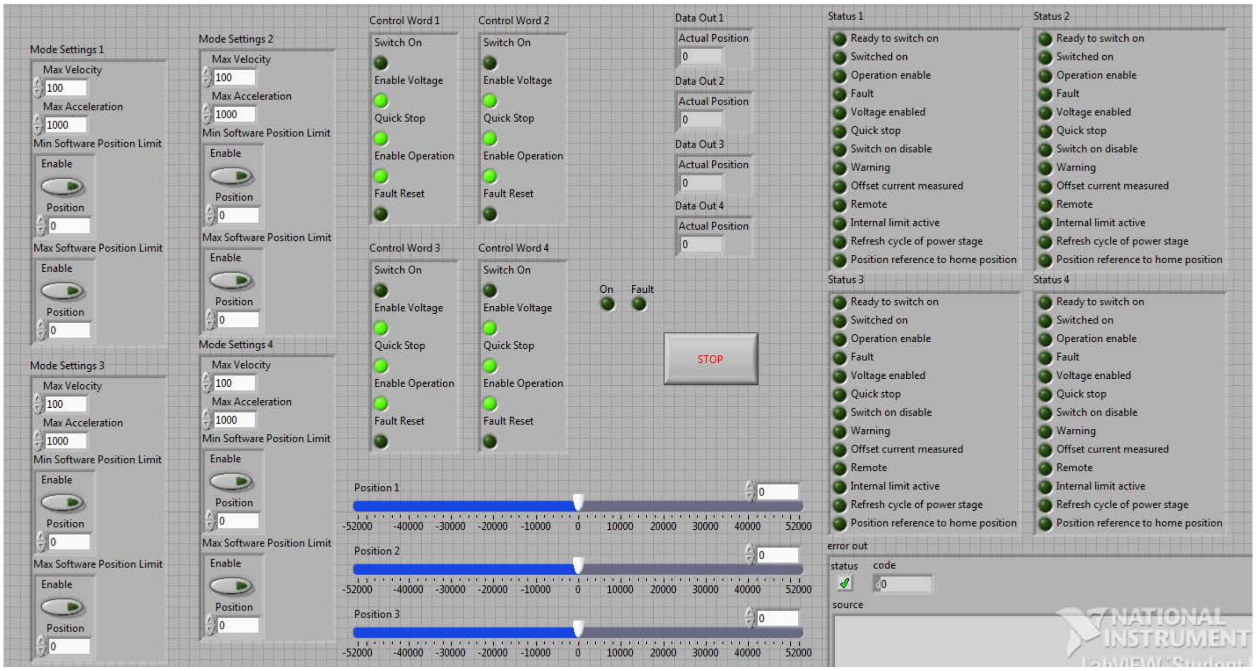


Figure 99: Networking

The networking layer starts a connection after 200ms with a remote host through a unique predetermined IP address given to each system. The IP addresses for the sbRIO and netbook are 10.0.0.1 and 10.0.0.2 respectively. As a connection is made, data is grabbed every 50ms.

UDP is a protocol for sending data without requiring any prior connections to be set up. This does not ensure proper delivery in the event of data corruption- when compared to TCP. Ensuring data alone does not matter to our system because by the time data is received its value has been compromised. We do not need this data because it is old now. The best thing for the system to do is wait for a new packet giving it a new position.

Another reason for choosing UDP is the simplicity in its architecture. All that is necessary is a source, data length, and data. Through testing our packets with our system architecture- all data was sent and not a single packet was lost in transmission.



Mode Settings 1
Max Velocity 100
Max Acceleration 1000
Min Software Position Limit: Enable, Position 0
Max Software Position Limit: Enable, Position 0

Mode Settings 2
Max Velocity 100
Max Acceleration 1000
Min Software Position Limit: Enable, Position 0
Max Software Position Limit: Enable, Position 0

Mode Settings 3
Max Velocity 100
Max Acceleration 1000
Min Software Position Limit: Enable, Position 0
Max Software Position Limit: Enable, Position 0

Mode Settings 4
Max Velocity 100
Max Acceleration 1000
Min Software Position Limit: Enable, Position 0
Max Software Position Limit: Enable, Position 0

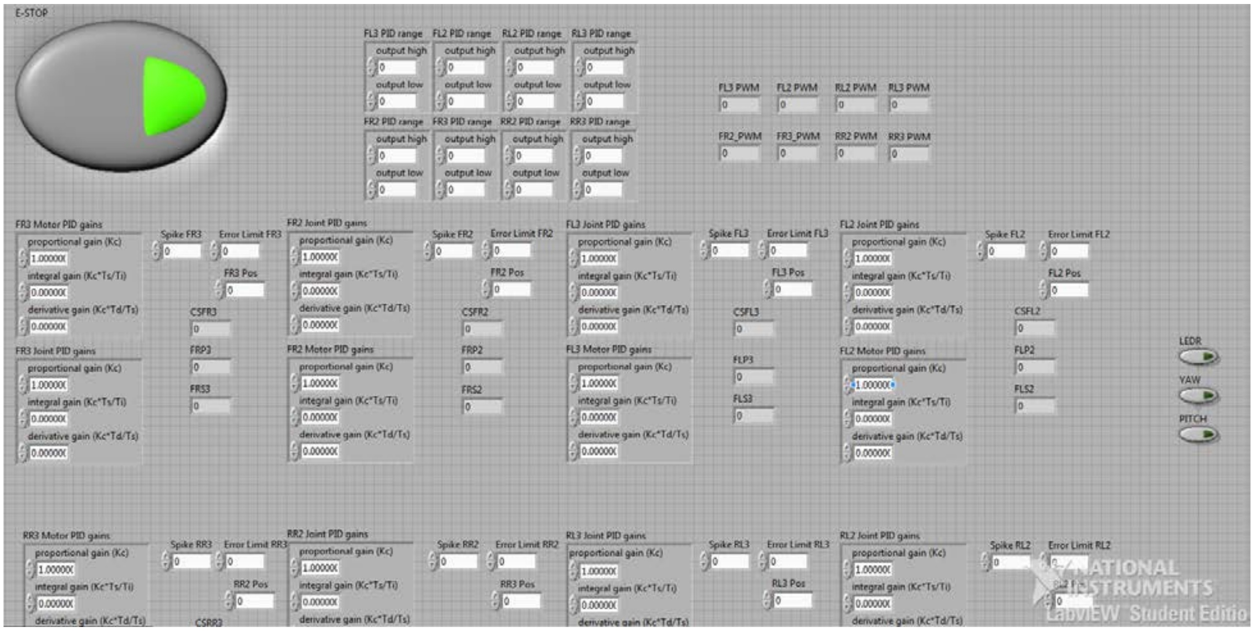
Data Out 1: Actual Position 0
Data Out 2: Actual Position 0
Data Out 3: Actual Position 0
Data Out 4: Actual Position 0

Position 1: 0
Position 2: 0
Position 3: 0

error out
status ✓ code 0
source

Figure 100 - EPOS2 Position Motor Control for 4 Motors

The Maxon EPOS2 library was found online through Maxon. This allowed for a quick start with many various features such as setting max velocity and max acceleration. When modified we were able to control all four motors of the hips. An additional modification had to be made for homing to work at initialization with limit switches. Homing is achieved through a control word send through UDP from the netbook. Errors and fault states could be monitored through the LabVIEW VI. However, because the final goal was to have full control of the sbRIO from the netbook, the error states are only available through the VI for testing and debugging purposes.



	FL3 PID range	FL2 PID range	RL2 PID range	RL3 PID range
output high	0	0	0	0
output low	0	0	0	0

	FR2 PID range	FR3 PID range	RR2 PID range	RR3 PID range
output high	0	0	0	0
output low	0	0	0	0

FL3 PWM	FL2 PWM	RL2 PWM	RL3 PWM
0	0	0	0

FR2_PWM	FR3_PWM	RR2 PWM	RR3 PWM
0	0	0	0

	FR3 Motor PID gains	FR2 Joint PID gains	FL3 Joint PID gains	FL2 Joint PID gains
proportional gain (Kc)	1.00000	1.00000	1.00000	1.00000
integral gain (Kc*Ts/Ti)	0.00000	0.00000	0.00000	0.00000
derivative gain (Kc*Td/Ts)	0.00000	0.00000	0.00000	0.00000

	FR3 Joint PID gains	FR2 Motor PID gains	FL3 Motor PID gains	FL2 Motor PID gains
proportional gain (Kc)	1.00000	1.00000	1.00000	1.00000
integral gain (Kc*Ts/Ti)	0.00000	0.00000	0.00000	0.00000
derivative gain (Kc*Td/Ts)	0.00000	0.00000	0.00000	0.00000

Spike FR3	Error Limit FR3	FR3 Pos	CSFR3	FRP3	FRS3
0	0	0	0	0	0

Spike FR2	Error Limit FR2	FR2 Pos	CSFR2	FRP2	FRS2
0	0	0	0	0	0

Spike FL3	Error Limit FL3	FL3 Pos	CSFL3	FLP3	FLS3
0	0	0	0	0	0

Spike FL2	Error Limit FL2	FL2 Pos	CSFL2	FLP2	FLS2
0	0	0	0	0	0

	RR3 Motor PID gains	RR2 Joint PID gains	RL3 Joint PID gains	RL2 Joint PID gains
proportional gain (Kc)	1.00000	1.00000	1.00000	1.00000
integral gain (Kc*Ts/Ti)	0.00000	0.00000	0.00000	0.00000
derivative gain (Kc*Td/Ts)				

Spike RR3	Error Limit RR3	RR2 Pos	CSRR3
0	0	0	

Spike RR2	Error Limit RR2	RR3 Pos
0	0	0

Spike RL3	Error Limit RL3	RL3 Pos
0	0	0

Spike RL2	Error Limit RL2	RL2 Pos
0	0	0

Figure 101 - Interface with Controls for all motors, ESTOP, and other testing features

This is the main interface with Sabertooth. This interface allows for PID tuning, velocity profiling, and full system monitoring. Displays stating the velocity being sent to each motor controller, every potentiometer input, and current through each motor controller are shown on this screen. A large software Emergency Stop button allows for each motor to be halted while communication with the system is available and doesn't require a full system reboot if triggered.

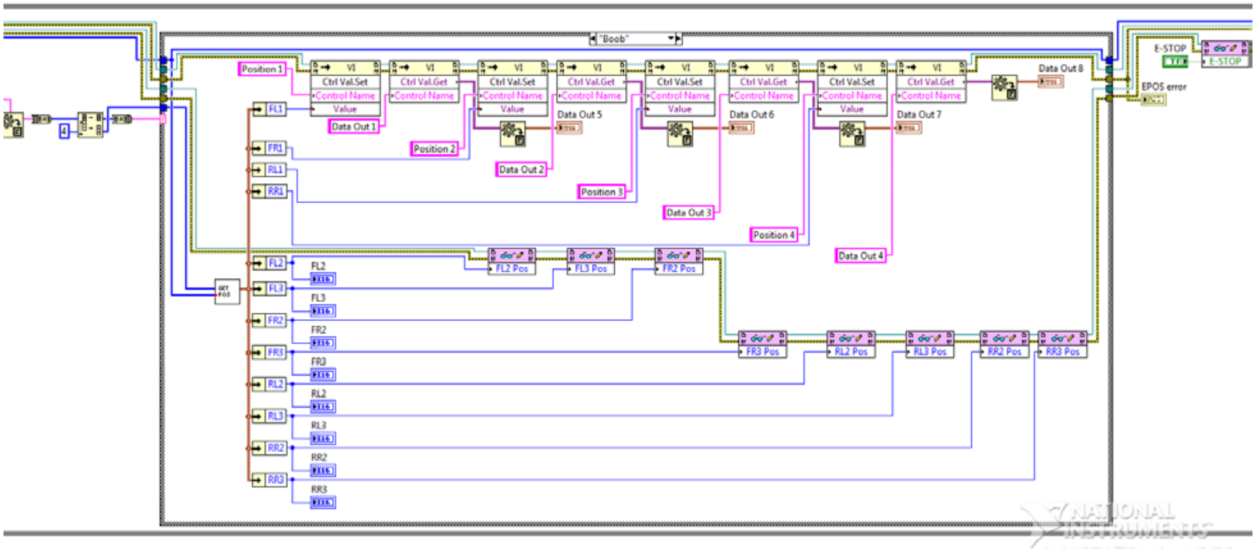


Figure 102 - FPGA Control Word and how it is utilized

Figure 102 is a screenshot of how a "block diagram" of LabVIEW code looks. This shows how the LabVIEW real-time processor parses through a UDP packet and sets variables in the memory to be used in later VI's such as motor control and FPGA PID controls.

LabVIEW was a necessary tool in the success and implementation of our Robotic system allowing high computational power within a powerful backend. The tools and libraries available from this industry standard allowed for integration within our system

3.5 Software Architecture

To interface the hardware components, the team developed an overarching program architecture responsible for managing the system. It was crucial to include the following aspects in this design:

- An interface for a user to control the robot remotely
- An internal database to store all information about the environment
- An interface for sending data to and receiving data from the microcontrollers
- An interface for planning future actions of the robot

With these criteria in mind, the following architecture was developed (Figure 103). The entire system is a two-process, multi-threaded application. The first process is used for all on-board control. It starts with the executive, which spawns off multiple threads for the other blocks involved in that process (the planner, knowledge base, sensors, and pilot/actuators). The second process is a Web-based user interface used to interact with a remote user.

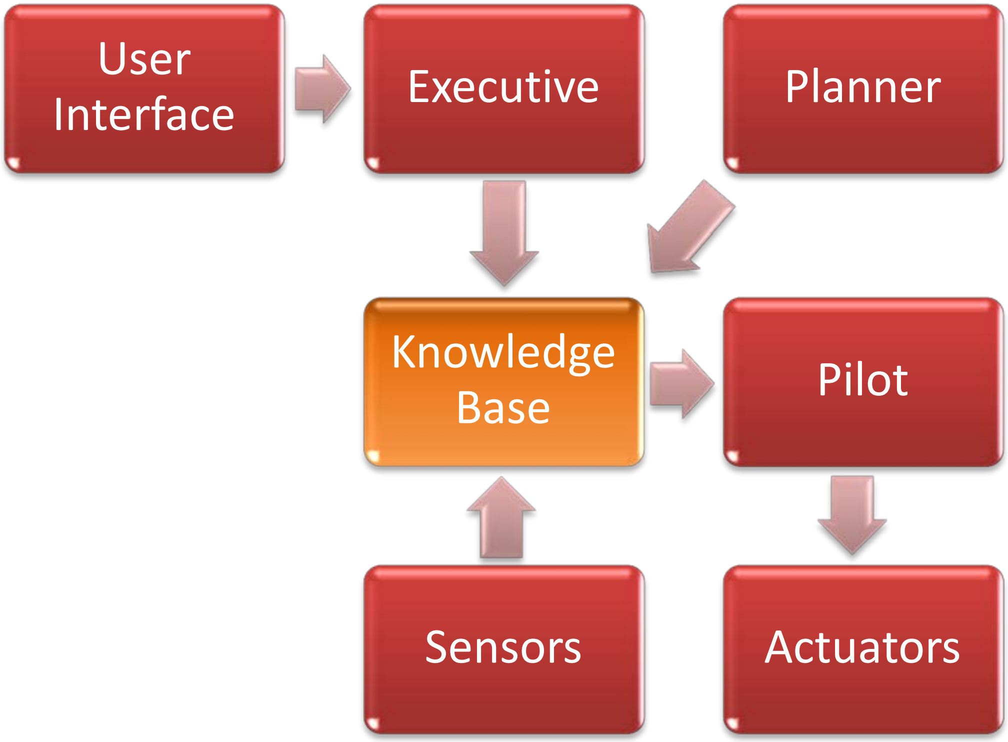
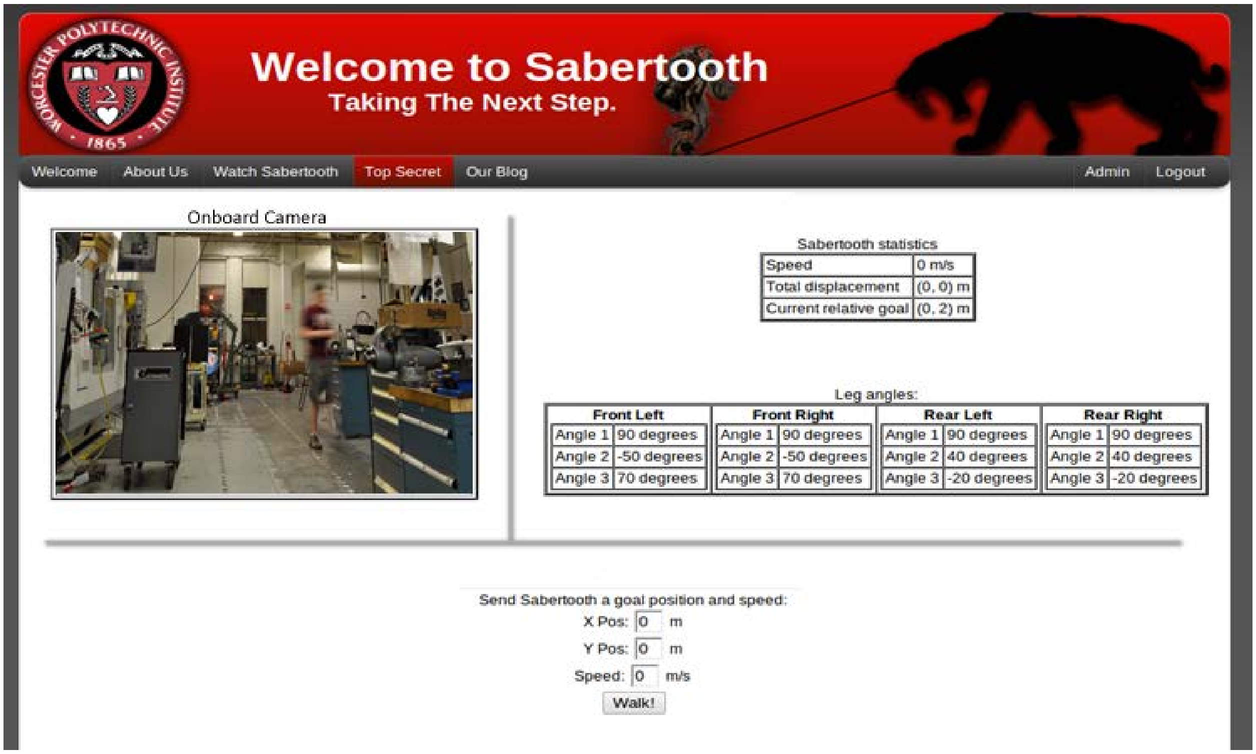


Figure 103 - Software architecture overview

3.5.1 User Interface

The user interface (UI) developed serves two purposes. First, it allows outside users to view the robot and monitor what it is doing and planning to do. The viewable data includes a live feed of the on-board camera, angular position data for each link, and Cartesian coordinates of where the robot is (relative to its starting position) as well as where it plans on going. Second, the UI allows for those with correct login information to control the robot. The user can input a Cartesian coordinate relative to the robot and a speed for the robot to travel at. The UI was constructed with the Turbo-Gears Python framework. The framework allows for the server to run on the on-board Eee PC and for any client with the IP address of the Eee PC to connect to the server through any browser (including desktop and mobile phone browsers). The UI communicates with the executive over a D-Bus inter-process communication protocol.



Welcome to Sabertooth
Taking The Next Step.

Onboard Camera

Sabertooth statistics

Speed	0 m/s
Total displacement	(0, 0) m
Current relative goal	(0, 2) m

Leg angles:

Front Left		Front Right		Rear Left		Rear Right	
Angle 1	90 degrees	Angle 1	90 degrees	Angle 1	90 degrees	Angle 1	90 degrees
Angle 2	-50 degrees	Angle 2	-50 degrees	Angle 2	40 degrees	Angle 2	40 degrees
Angle 3	70 degrees	Angle 3	70 degrees	Angle 3	-20 degrees	Angle 3	-20 degrees

Send Sabertooth a goal position and speed:
X Pos: 0 m
Y Pos: 0 m
Speed: 0 m/s

Figure 104 - User Interface

3.5.2 Executive

The executive is used to control the entire software system. It notifies all other components of the architecture when to activate, and controls when incoming data is given attention. This aspect of the executive is broken into five main sub-responsibilities. First, it initializes the entire system by creating the threads involved in the process. This step also entails connecting to the Web-server. The sensors thread is then notified via flags in the knowledge base that it should start recording data about the world, including the three-dimensional map of the environment. The data is stored in the knowledge base. Next, the planner is notified to plan a path through the environment. This is dependent upon input from the user interface, so until that input is given, the process blocks. Once that input is received, the path can be planned. Next, the executive notifies the planner to calculate the gaits for walking. Once all gaits are calculated and modified, the executive notifies the pilot to send the data to the actuators. This cycle repeats, until one of three things happens: a new input is provided from the user interface, the robot reaches the goal, or an error occurs (such as an impossible goal, battery power runs out, or lost communication). If an error occurs, the executive takes all permissions away from the other blocks and exits, so nothing is sent to the robot.

The following diagram depicts the general overview of the executive's tasks:

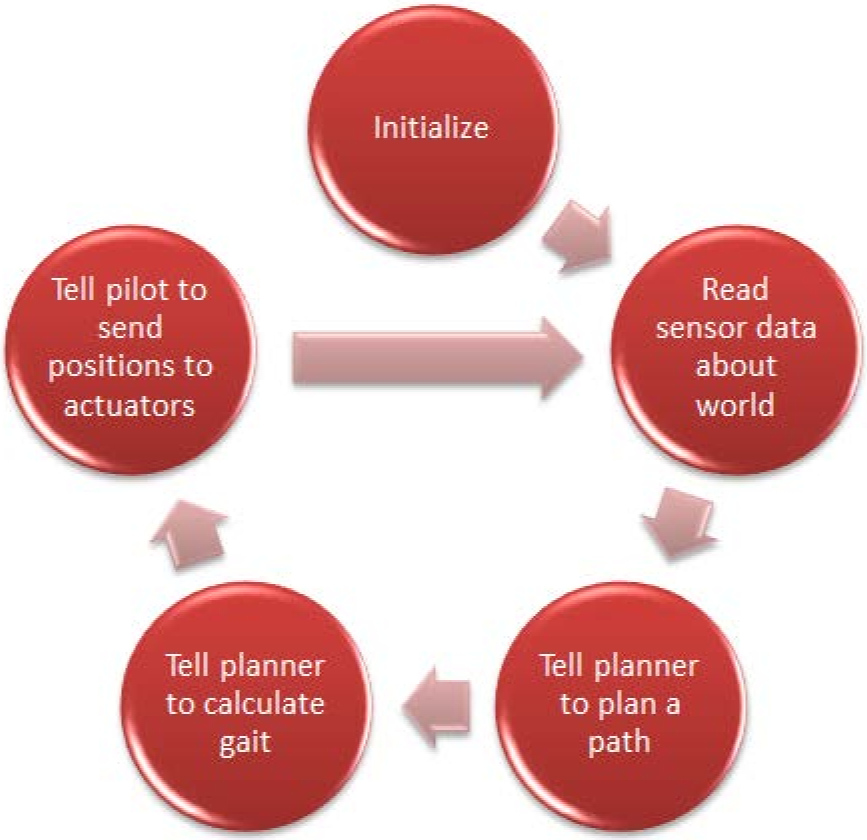

Initialize
Read sensor data about world
Tell planner to plan a path
Tell planner to calculate gait
Tell pilot to send positions to actuators

Figure 105 - Executive overview

3.5.3 Sensors

The sensors block is simply responsible for reading sensor data from the SBRIO and storing it in the knowledge base. While the sensor block is constantly polling for data, the executive controls when the sensor block actually stores the data in the knowledge base. This solves the problem of different sampling rates – the data from each sensor comes on at different rates, being sent to the sensor block asynchronous with one another. By storing data only when instructed by the executive, all sensor data is reported to the rest of the program synchronously.

3.5.4 Planner

The planner has two main responsibilities – planning a path and planning gaits. To plan a path, the planner executes the following steps:

1. Read three-dimensional map from knowledge base
2. Filter the map for noise
3. Compress the map into a two-dimensional occupancy grid
4. Expand occupancy grid with C-Spacing
5. Plan the path

The executive thread is blocked until notified by the executive that the three-dimensional map is ready. At this point, a low pass filter is applied to the map to reduce noise from the LIDAR. The low pass filter works by taking in a tolerance and a filter size as parameters. The tolerance is a cutoff measure used to decide if a data point should be marked as noise. The filter size is the size of the rectangle passed over the map that applies the filter inside that rectangle. Essentially, this rectangle is moved across the whole map and the data points in that rectangle are compared with one another with regards to the tolerance.

Next, the three dimensional map is compressed into a two-dimensional occupancy grid. To do this, the team first made the assumption that, for testing purposes, the robot can assume it has height clearance. This allows for treating the occupancy grid as a top-down view of the three-dimensional map as shown below in Figure 106 and Figure 107:

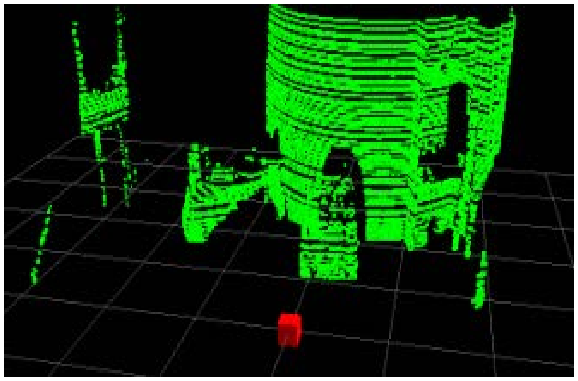


Figure 106 - Three-dimensional map

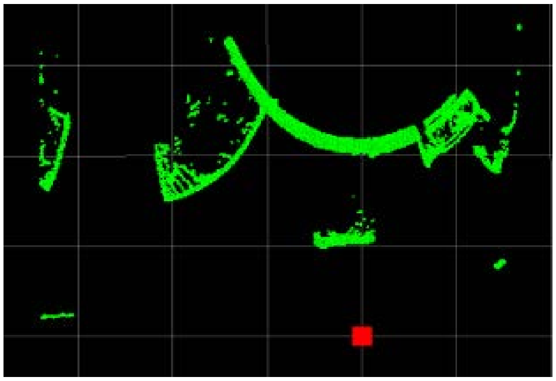


Figure 107 - Top-down view of 3D map

The size of each square of the occupancy grid is 10cm by 10cm. Given the dimensions of the robot, the grid size is small enough to be accurate, and large enough to reduce computational resources required to compress into squares of smaller size. The empty occupancy grid was then iterated through and each square becomes marked as occupied if a corresponding point in the range of 10cm on the three-dimensional map has a distance greater than the maximum range of the LIDAR (meaning that there is an object occupying that point in space). Using the three-dimensional map above, the following occupancy grid was constructed (as can be seen, it looks very similar to the top-down view of the three-dimensional map):

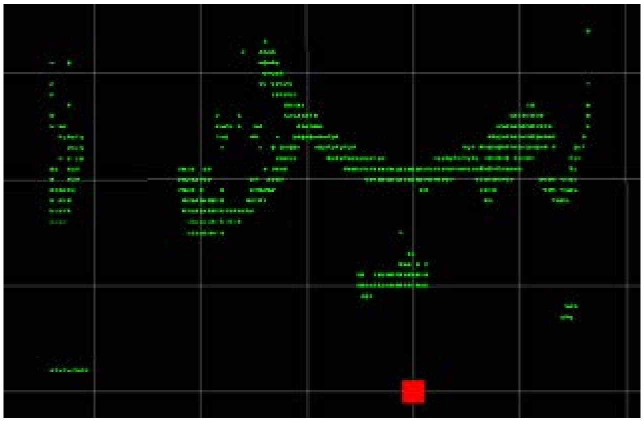


Figure 108 - Occupancy grid

The next step is to account for the dimensions of the robot.Since path planning algorithms assume the point moving through space is only the size of that one point, the algorithm has to be modified to account for the robot's size. This could be done in two ways: first, that path planning algorithm could be modified to ensure that all cells within a certain radius of an unexplored node are unoccupied before exploring it; second, the occupancy grid could be modified to account for the dimensions of the robot. The team implemented the latter with a technique known as generating a C-Space. Since the point moving through space in the algorithm represents the middle of the front of the robot (where the LIDAR is located), the length of the robot can be disregarded (assuming the robot will be moving forward). Each point on the occupancy grid can then be expanded by half the width of the robot (i.e. all points within that radius plus some tolerance are also marked as occupied). This ensures that at all times, the point moving through space is at least half the width of the robot away from the actual obstacle.

Once the C-Spacing is complete, the final step is to plan a path. The team decided to implement an A* path planning algorithm to perform this task. Dijkstra was disregarded due to the extra computation time spent calculating unneeded data, and A* was chosen over D* because the added benefits of D* do not apply to this application. Running A* on the previous occupancy grid (after expansion) resulted in the following path to an arbitrary point on the opposite side of the map:

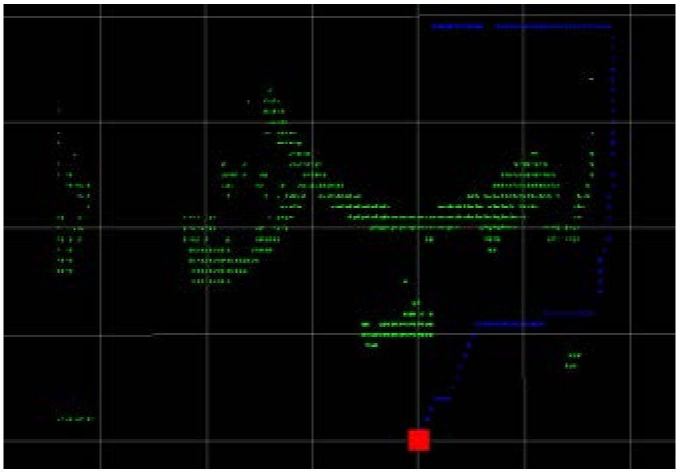


Figure 109 - Occupancy grid with path

After the path is planned, the path is stored in the knowledge base. The executive then validates the path (by making sure a path was found), and then informs the planner to perform gait calculations. Further explanation of the gait calculations can be seen in Section 3.1.5 Gait Generation. The following diagram depicts the general overview of the planner's tasks:

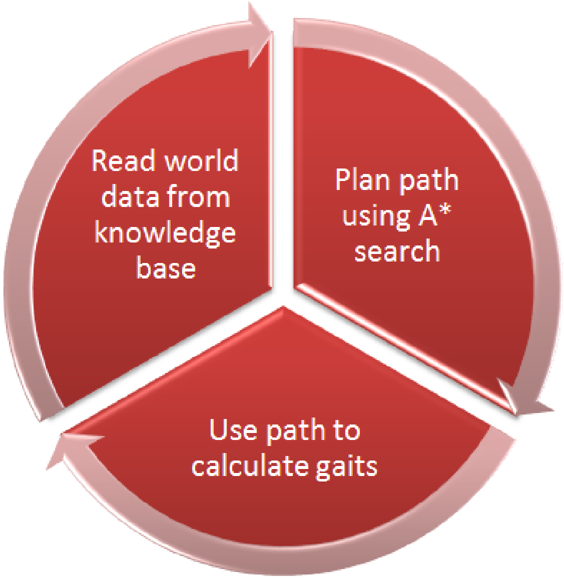

Read world data from knowledge base
Plan path using A* search
Use path to calculate gaits

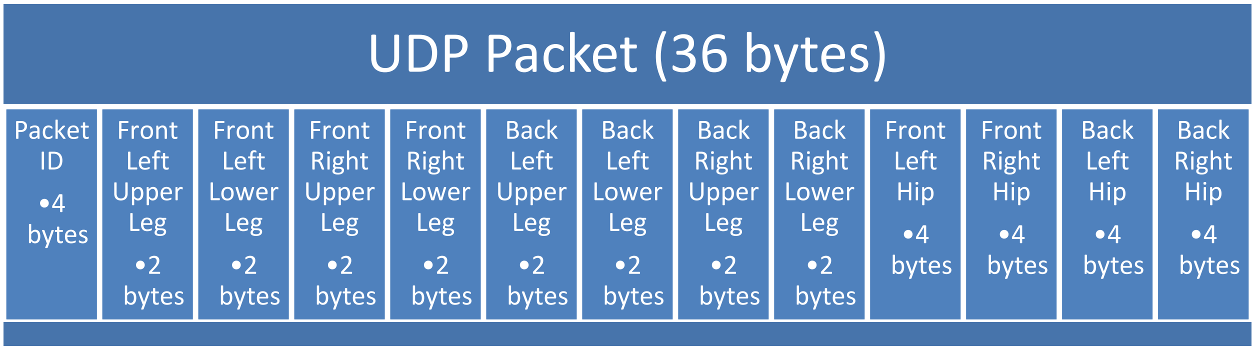
Figure 110 - Planner overview

3.5.5 Pilot and Actuators

The pilot's only responsibility is to send the gait calculations to the SBRIO in the form of potentiometer values. The gait calculations only provide the angle (in degrees) of each link. To convert this to potentiometer values, the following formula is used:

$$potVal = \frac{potHigh - potLow}{angleHigh - angleLow} * angleVal + potLow - angleLow\left(\frac{potHigh - potLow}{angleHigh - angleLow}\right)$$

In the formula, potHigh refers to the potentiometer value at the links highest point, and potLow refers to the opposite. The same convention was used for angleHigh and angleLow. These values are sent to the actuators in a UDP packet at a rate of 50 milliseconds, where the packet is defined as in Figure 111.



UDP Packet (36 bytes)

Packet ID	Front Left Upper Leg	Front Left Lower Leg	Front Right Upper Leg	Front Right Lower Leg	Back Left Upper Leg	Back Left Lower Leg	Back Right Upper Leg	Back Right Lower Leg	Front Left Hip	Front Right Hip	Back Left Hip	Back Right Hip
•4 bytes	•2 bytes	•2 bytes	•2 bytes	•2 bytes	•2 bytes	•2 bytes	•2 bytes	•2 bytes	•4 bytes	•4 bytes	•4 bytes	•4 bytes

Figure 111 - UDP Packet Layout

Figure 112 depicts the general overview of the pilot's tasks:

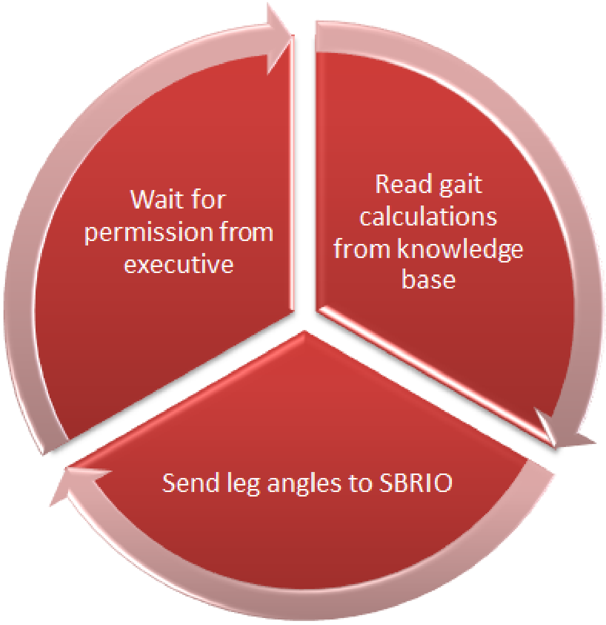

Wait for permission from executive
Read gait calculations from knowledge base
Send leg angles to SBRIO

Figure 112 - Pilot overview

3.5.6 Knowledge Base

The knowledge base stores all information about the world. This data includes the current three-dimensional map of the environment, the two-dimensional occupancy grid, the most recently planned path, and the most recently planned series of gaits. Everything in the knowledge base is semaphore protected because the process is a multi-threaded application – each semaphore allows only single access to the data. If a thread tries to access some data while another thread is already accessing it, the thread will block until it is available again.

The knowledge base also stores a series of flags that act as permission granters. They are used by the executive to notify another thread when it should be doing something. If an error occurs, every flag is disabled so no thread has permission to execute anything.

CHAPTER 4: RESULTS AND DISCUSSION

4.0 Mechanical Overview

Following the team's completion of the robot a full mechanical overview was developed to allow for a well-defined knowledge base of the robot's capabilities. A picture of the robot was also taken and can be seen in Figure 113. The full table of the robot's capabilities can be seen in

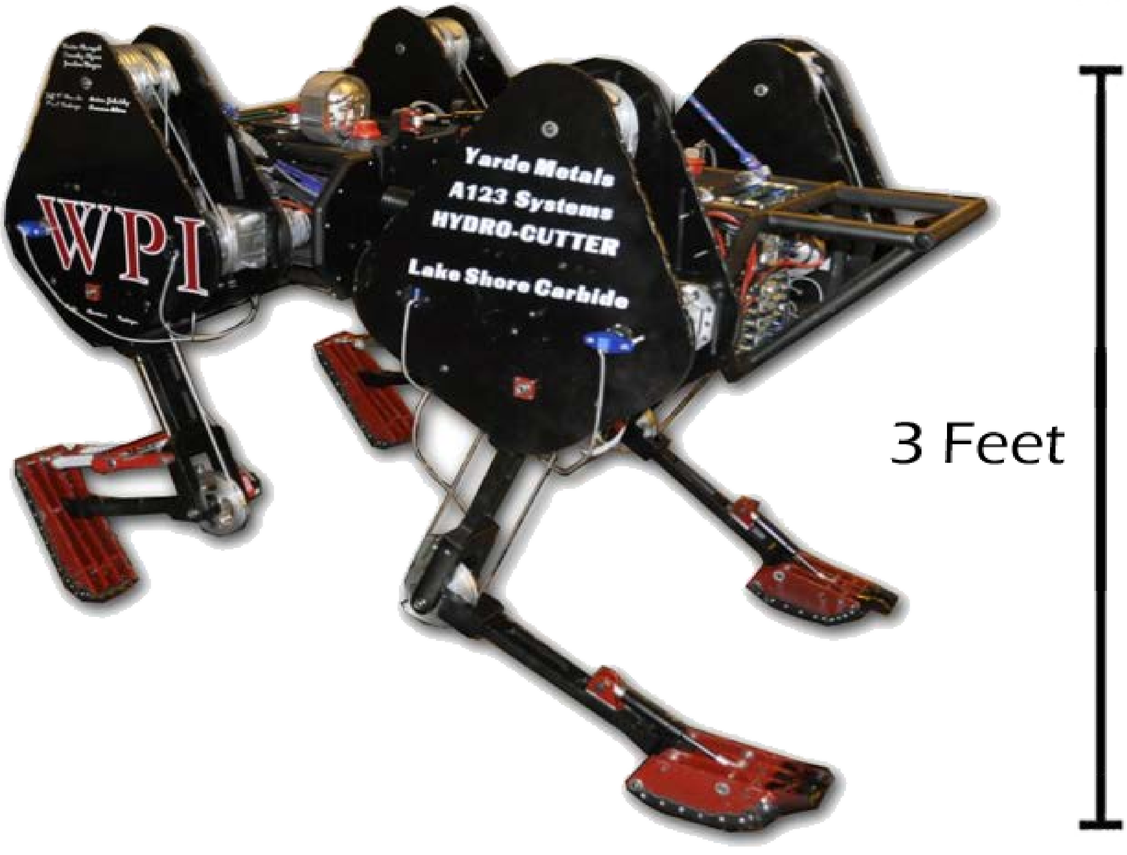


Figure 113 - Sabertooth's Final Model

Robot Statistic	Value
Robot Width	34”
Robot Height	36”
Robot Length	49”
Robot Weight	375lb
Maximum Theoretical Speed	5ft/sec
Maximum Foot Stride	15”
Maximum Theoretical Payload Capacity	30lb
Maximum Foot Camber	±10 degrees
Body Joint Roll	50 degrees maximum (based on body joint yaw)
Body Joint Yaw	10 degrees maximum (based on body joint roll)
System Power Capacity	27.6Ah
System Voltage	28.8V for motors, 5V for sensors
Position Sensors (per leg)	4 Potentiometers, 1 at knee joint, 1 at shoulder joint, 1 at knee motor, 1 at shoulder motor 1 incremental encoder at every hip motor

Table 20 - Robot Overview

4.0.1 Frame Assembly

The initial design the team attempted to assemble for the frame sections involved bending tubing for several of the pieces. This was attempted using manual tube bender like the one in Figure 114. After several days of research and many practice attempts, it was determined this process was unsuitable for creating the frame.

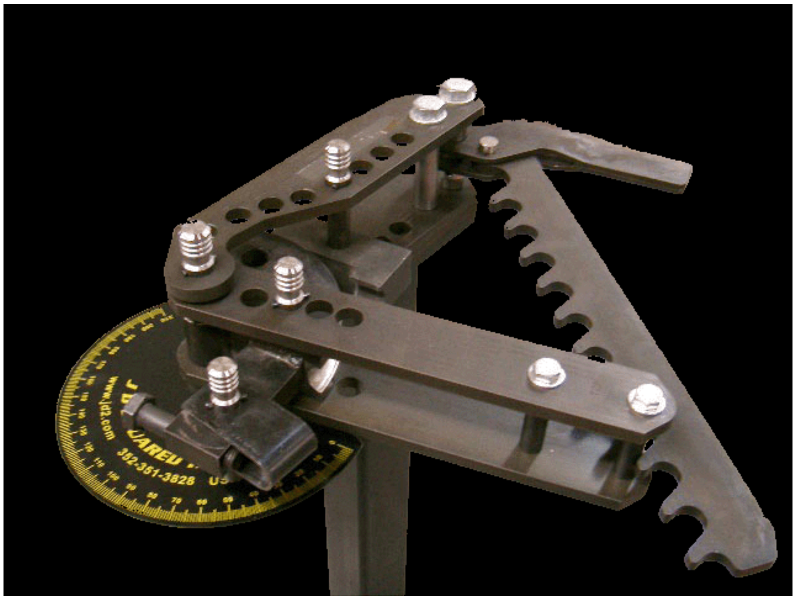


Figure 114 - Manual Tube Bender

One of the major problems was that the tube kinked on the inner radius if bend beyond about 30°, as seen in Figure 115. We attempted to solve this issue by packing the tubing with fine sand before bending, but the same results occurred.

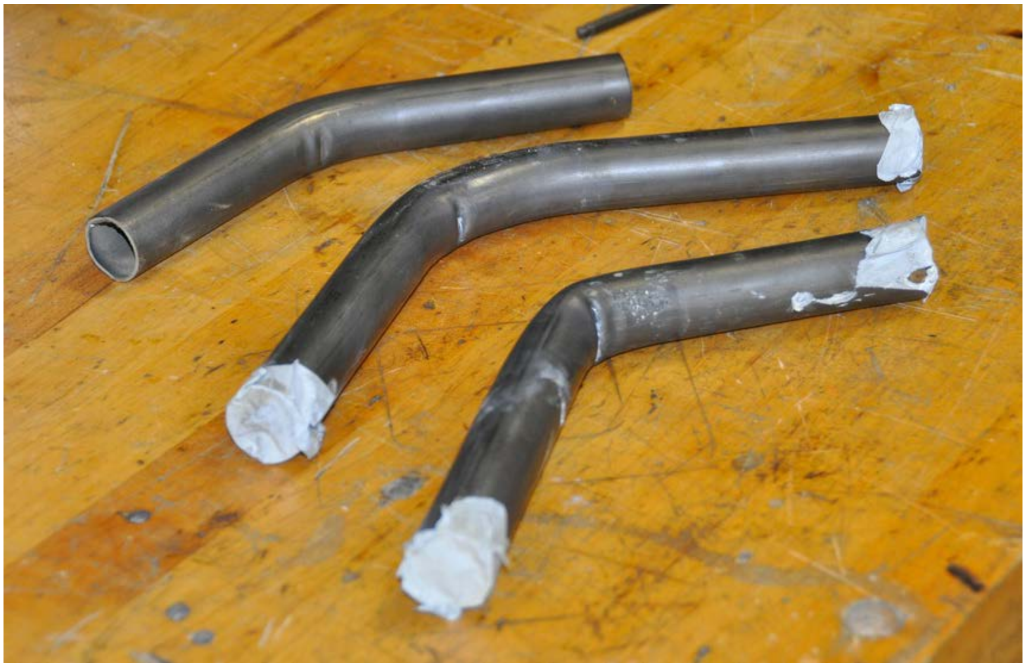


Figure 115 - Example of Kinked tubes

Because of this, the frame was slightly redesigned, with each bend being replaced with an angle cut and a weld. These parts were then fabricated in preparation for welding. The welding process also was more difficult and took longer than was originally anticipated. This was because tight tolerances had to be held over long distances. Additionally, the piece had to be set up and then carefully welded in place without moving other parts of the frame. Figure 116 shows a welded frame section.

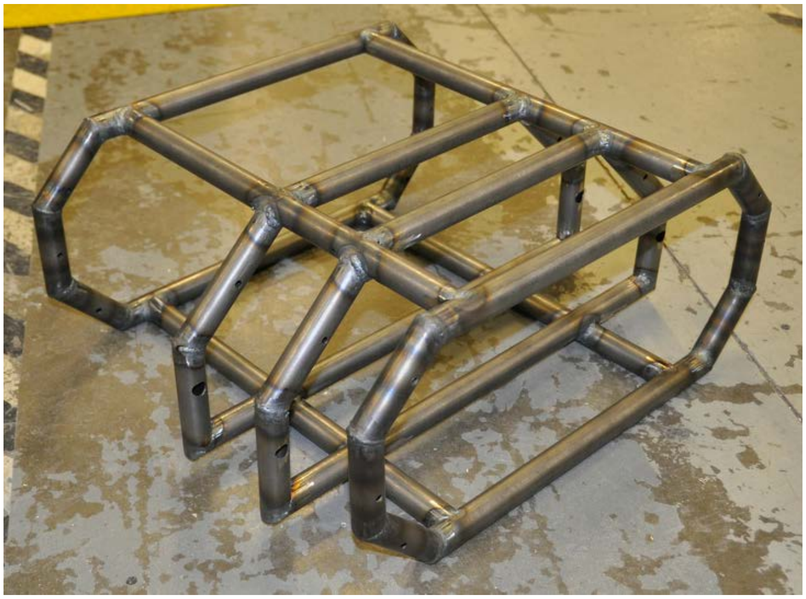


Figure 116- Welded Frame Section

Overall though the frame functioned as intended, with component mounting to it with little difficulty, no complications from tolerances, and a strong rigid protector for the internal power systems.

4.0.2 Hip Assembly

Being that the first actual linear actuator produced was to be essentially the prototype it was the teams priority to have it be the first subsystem assembled. The first part of the system assembled was the mounting bearings, lead screw, and push rods to test by hand for stiffness and friction. Slight adjustment to ensure the straightness of the bushings the push rods were riding in were required, and then small misalignments were dealt with by sliding the push rods back and forth within the bushings to wear into a smooth fit. Once this was confirmed the rest of the actuator was assembled as shown in Figure 117.

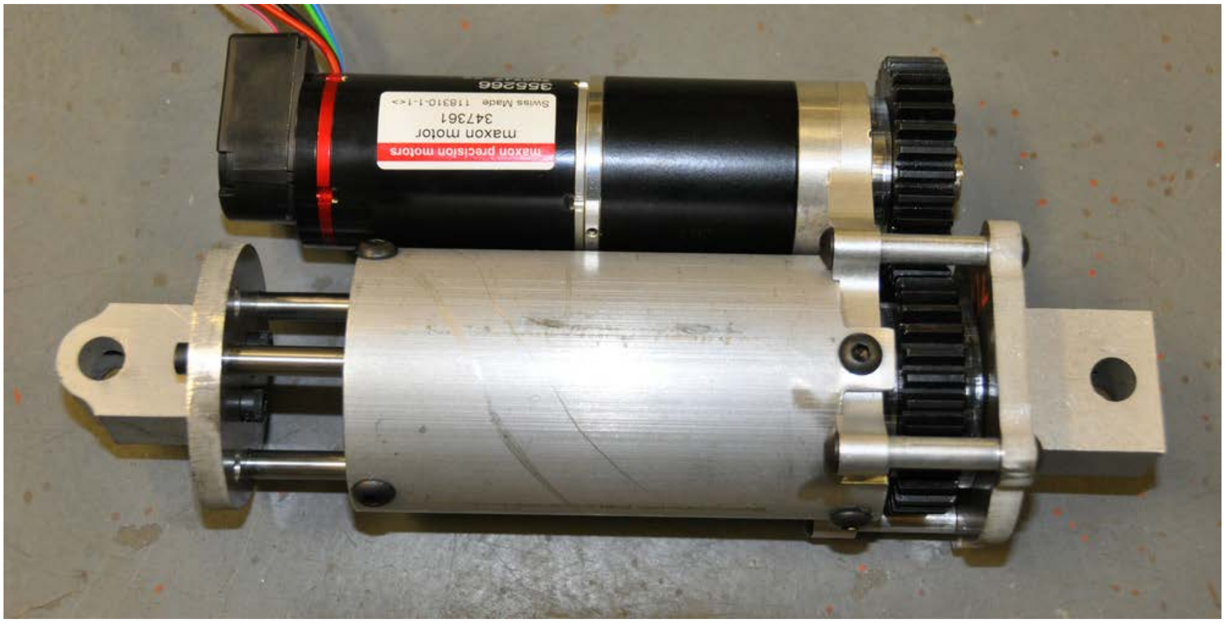


Figure 117- Completed Linear Actuator

The actuator was then hooked up to a Maxon speed controller getting power from an external power supply. The actuator was run from on limit to the other and timed. It took approximately 3.5 seconds to travel the entire distance, only 0.5 seconds off the calculated speed. Additionally, the current draw on the motor was approximately one amp throughout the test meaning there was no binding or unexpected friction in the system. With this knowledge the team felt confident enough in the design to create 3 more actuators.

The second part of the hip system was the assembly of the actual leg interface with the frame as seen in Figure 118. This part of the assembly interfaced nicely with the frame and was able to support the weight of the robot during standing tests. One problem was discovered though; the pivot that attached to the leg plate had a small amount of play on the fit with the shaft. This amplified over several inches of leg plate meant that the leg had a small wobble.

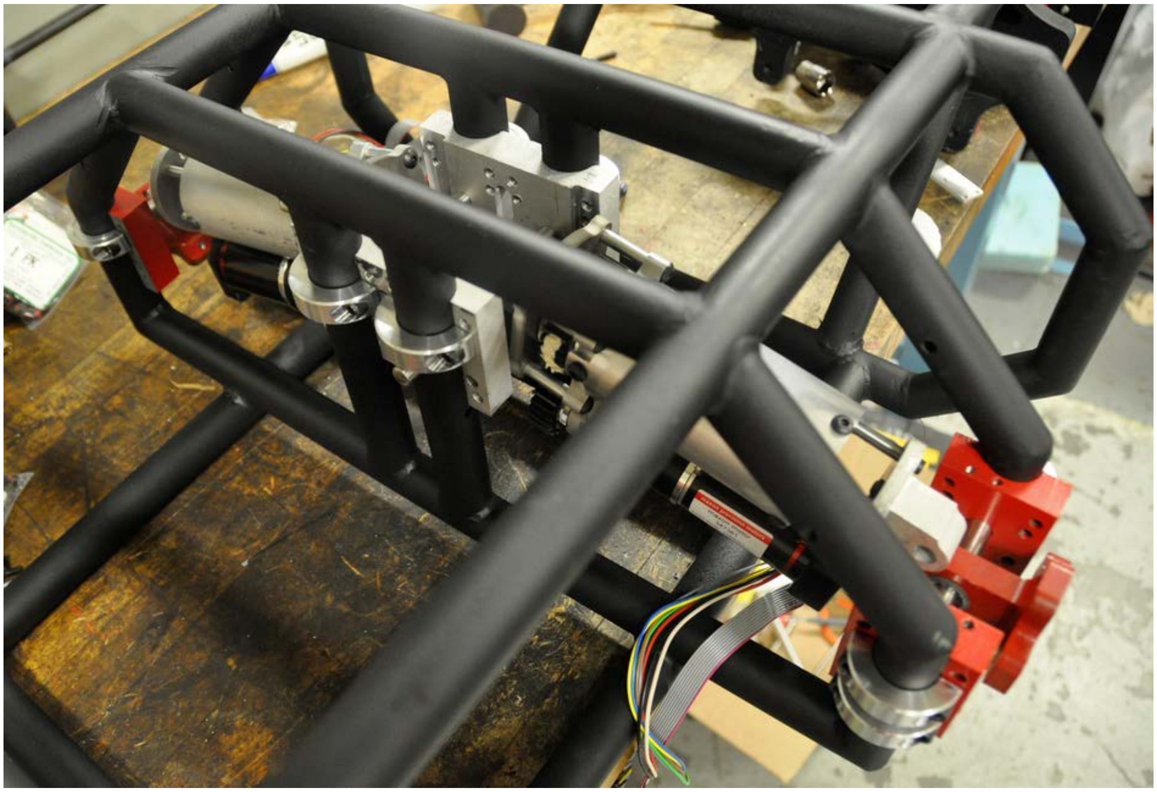


Figure 118- Full Hip Assembly in Body Section

The team tried to fix the problem by first adding shaft collars to either side of the pivot mount in order to stop the rocking. This helped a bit, but there was still approximately a quarter inch of rock at the edge of the leg plates. A solution developed to further reduce this rocking was to insert brass bushing in the pivot block to make the fit on the shaft tighter. While this stopped the rocking, after a few days of wear the bushings flared out and the legs began to rock again.

4.0.3 Body Joint Assembly

After assembly of the final prototype the team proceeded to make a final version of the body joint assembly for assembly on the robot. The final version of the body used all machined aluminum components coupled with commercially available gas springs and commercially available shoulder bolts. To machine the pieces, the team first rough waterjet cut the parts required for the full assembly. Upon receiving the parts, the team then CNC milled the two end plates, the 2 large bushing blocks and the 8 mini u-joints. The most complex part of the CNC machining was indicating the pieces to be square in the machine such that the different setups between the waterjet machining and the multiple setups in the mill did not stack up and cause an unusable part. Specific care was taken to indicate features of the same corners between different setups.

Upon assembly of the body joint, the team quickly realized that there would be two serious issues with the proposed design. The first issue was the pre-bias in the gas springs. Due to the way the springs were positioned, the body joint would have a significant bias to be off skew in the yaw rotation. After careful analysis the team concluded the reason was because the moments on the system were not balanced in the configuration that the gas springs were originally placed. Luckily, however, this was a simple fix of moving around the location of two of the gas springs. By imposing this quick change the body joint was perfectly balanced and preferred to skew towards the center. The team did notice this as a possible problem with the original prototype but attributed it towards poor manufacturing tolerances. Luckily, there was a very simple fix. Additionally, much as expected, since the transmission angles of the system changed, despite using constant force gas springs, as we went through the motion of a rotation the further we went the harder it would be to continue going through the motion. This would be perfect for our system since we would want it to resist change in a non-constant manner similar to other quadruped spines (Gray, 1932).

The other problem resulted in the force of the selection of the gas springs. When building the prototype model of the body joint, due to the extreme weak nature of all the materials used the 15lb gas springs appeared to be very strong for the system. However, once the 15lb gas springs were transferred to the actual system, they appeared to provide a negligible amount of resistance almost as though they were not even in the system. Luckily, this too was an easy fix simply replacing them with higher force gas springs. The team chose to replace them with 60lb gas springs. This too, however, was simply a guess by what we believed to be a correct stiffness. Looking back at the original task specification the team succeeded in developing a 2 degree of freedom body joint separating the two halves of the robot. A final picture of the completed body joint design can be seen in Figure 119.

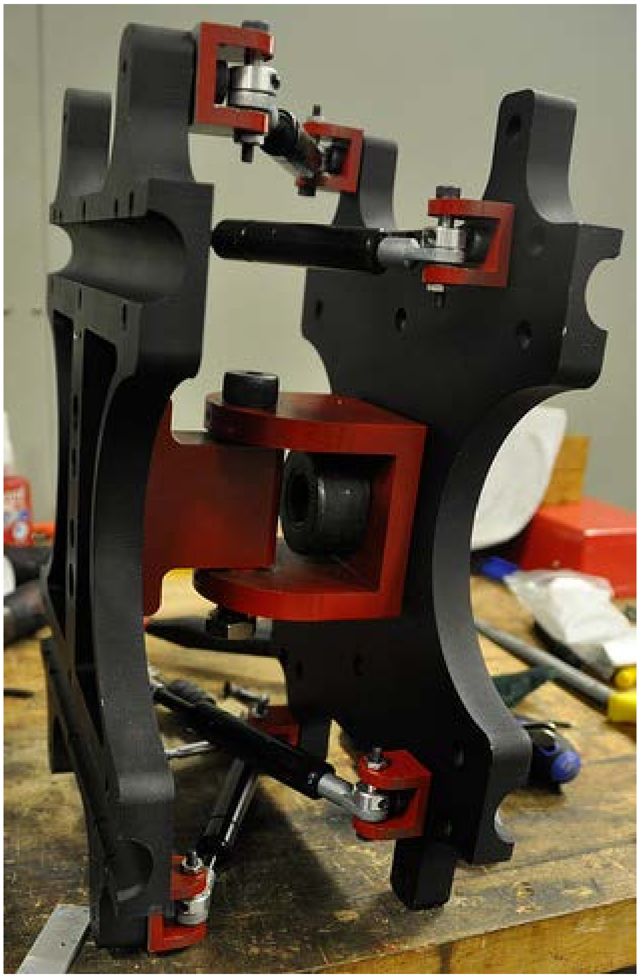


Figure 119 - Completed Body Joint

4.0.4 Sensor Head Assembly

The final sensor head assembly was constructed out of acrylic and ABS. The final design resulted in no problems. Each component that the sensor head was made for fit appropriately, and the range of motion of the tilting and panning servos was unobstructed. More importantly, the LIDAR's scanner is able to complete a full scan without any parts of the sensor head or the robot hindering the scanner. The sensor head also protects the LIDAR from damage two ways: first, LIDAR is on top of the sensor head, so for the LIDAR to make contact with the ground, the robot would have to roll 180 degrees on its side; second, it is surrounded by ABS on all sides that don't block the scanner. The final sensor head assembly mounted on the front of the robot is shown in Figure 121.

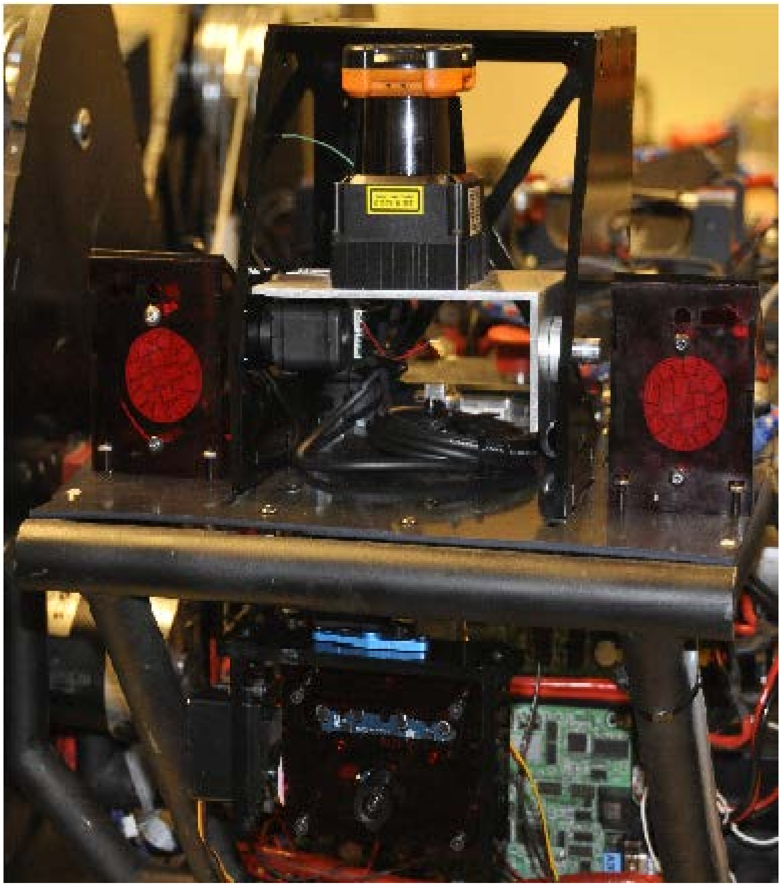


Figure 120 - Completed Sensor Head

4.0.5 Spring Pod Assembly

The spring pods proved to be one of the most frustrating systems to implement. The spring pods were meant to compress the spring when pulling the two ball and shank fittings. Manufacturing the system proved to be exceedingly difficult due to the spring tolerances. While the springs were commercially bought they did not come in the size shown on the downloaded CAD model. Rather, the spring for the shoulder had a rod diameter of roughly 1/16" larger than shown in the model. Since the aluminum tubes specified for this application were already very thin removing an extra 1/16" of the full wall caused the tubes to retain only a .050" wall. This tolerance made the tubes very weak as a whole and have a jagged bore on the inside due to the very thin wall machining that needed to be performed to hone the inside of the tubes. Machining the plungers and hard stops on a lathe resulted in little to no issues.

The team ran into serious issues with the ball and shank fittings. Upon receiving the fittings the team quickly realized that there would be problems swaging the end of the cable onto the fitting. The proper way to swage the cable would be to use a hydraulic press with the appropriate die. Unfortunately, the team did not have access to such a machine and hence tested various ways to replicate such swaging techniques with a traditional Nicropress hand swaging. Despite all the team's best efforts the tool simply did not handle the job and even applying a minimal amount of force on the ball and shank fitting would cause the wire rope to pull out. Simply put, the team did not use the right tool for the job. Once further testing had been completed, the team proceeded to search for a local company which would be able to professionally machine swage the ball and shank fittings. The team was able to contact St. Pierre, a local Worcester company, who agreed to help with the project and used a 20 ton hydraulic press with proper die to swage the ball and shank fittings with the proper tool. Additionally, St. Pierre was able to validate that our connectors were properly swaged by using a hydraulic tensile testing tool to pull on the two ends of the cable with over 3000lbs of force. At 3000 lbs the connectors did not pull off the stud.   A picture of the tensile testing setup can be seen in Figure 121.

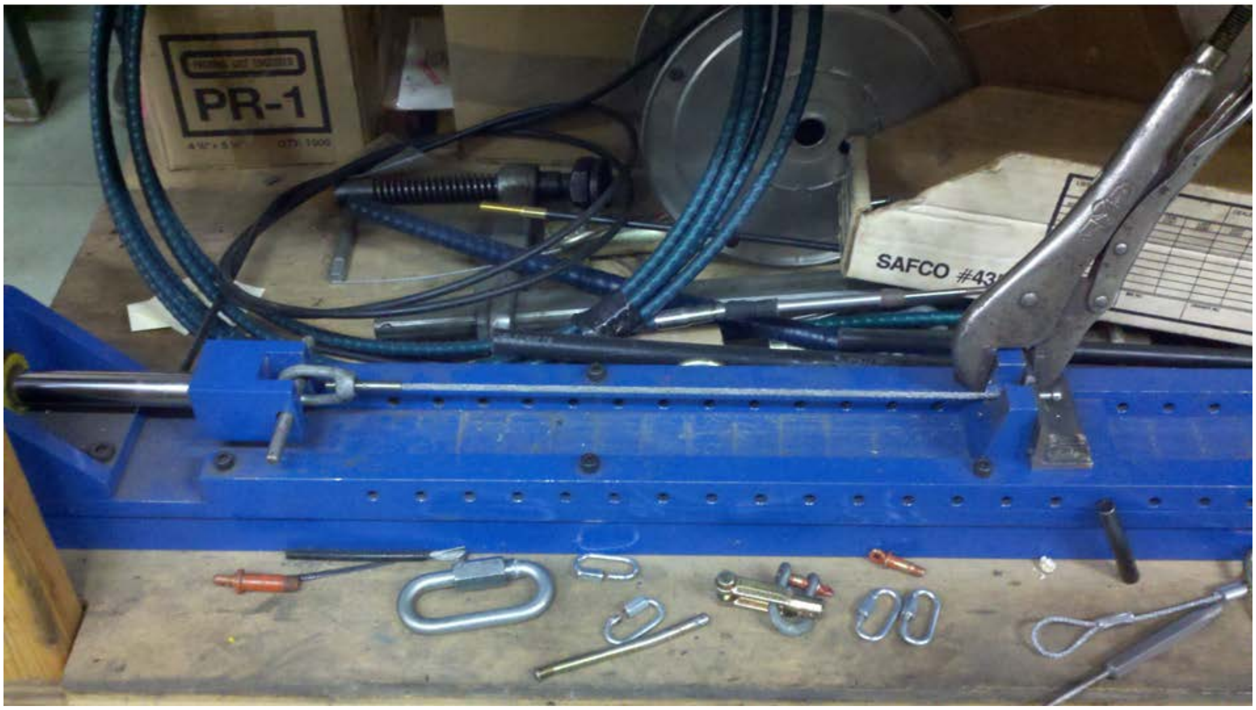


Figure 121 - Tensile Testing Setup

Once solving that problem the spring pods were fully functional. The shoulder spring pod was still not ideal due to the extremely thin wall, however never failed during any testing. These spring pods were the basis of the compliant spring system which would allow the robot to reproduce a SLIP model. A picture of a few of the disassembled and assembled spring pods can be seen in Figure 122.

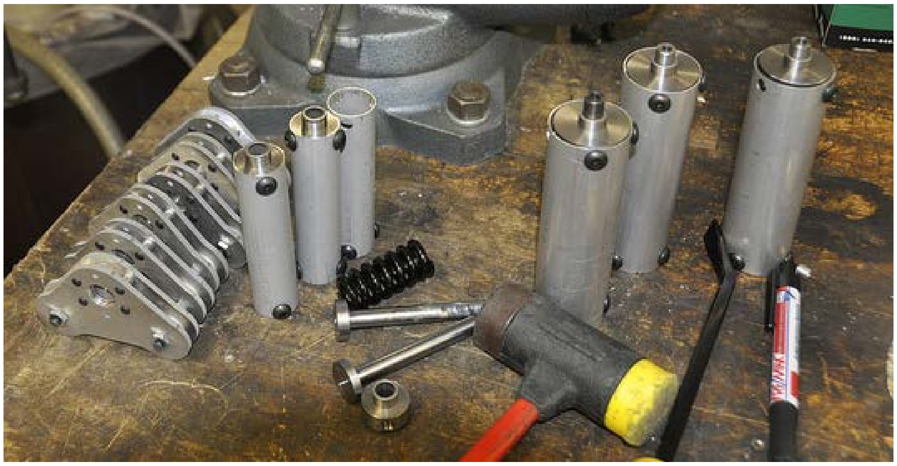


Figure 122 – Assembled Spring Pods

4.0.6 Cable System Assembly

The cable system proved to be the hardest system to assemble. The team began by first laying out all of the capstons and determining their proper location on the shoulder plates. Having four different capstons which all looked extremely similar led to great frustration. This problem was quickly overcome by punching letters into each unique pulley. A was punched into the capstons coinciding with the red color in the CAD model, M coinciding with the blue capstons in the CAD, O coinciding with the pink capstons and the motor capstons were left unmarked. Once laid out the necessary capstons were pinned together with 3/16” dowel pins. Necessary spacing between the two capstons was set off using nylon spacers. Once set off the dowel pins were pressed in and secured with green Loctite. Next the team set the capstons on the shaft and inset the shafts into the shoulder plates. Once set, the team then began measuring cable lengths and cutting wire rope to the proper length. While the team had calculated theoretical cable lengths, upon quick inspection it became clear that simply measuring on the actual physical system would be much more accurate. After measuring and cutting all the cables the team began stringing the cables through each capston. Similarly to the issues with the spring pods the traditional threaded stud connectors had issues with cables pulling out during testing. The solution, much like the ball and shank stops for the spring pods, was bringing the stud connectors to a professional swaging place. Once professionally swaged the cables were never burdened by the issue of tearing out of the studs.

Once fully assembled the team was able to use the passive tensioners to properly tension the cables. Over time each of the cables would need re-tensioning. A full picture looking into the assembled cable system can be seen in Figure 123.

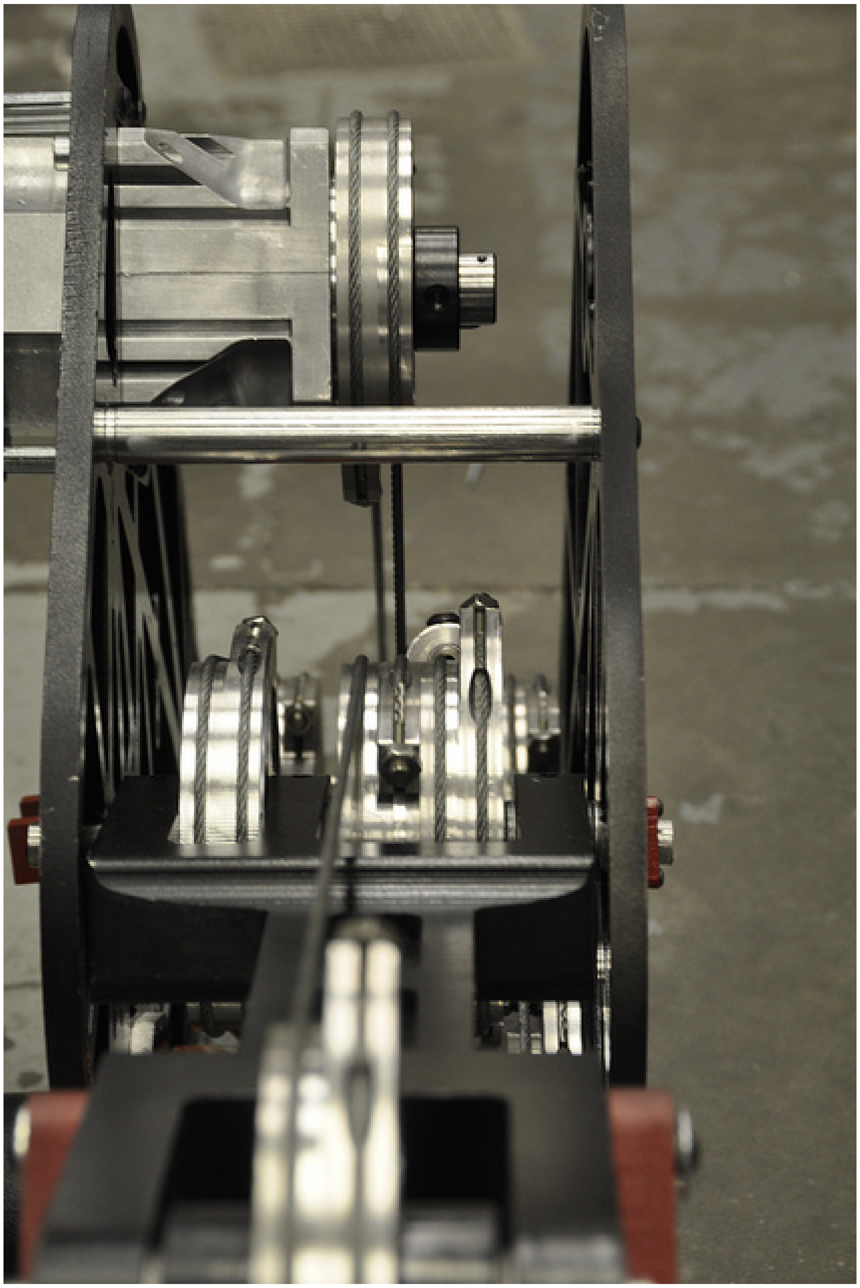


Figure 123 - Fully Assembled Cable System

The team still ran into a number of issues with cables interfering into various parts of the system. These were solved by carefully rerouting the cable such that all of the interferences were no longer an issue. This involved slightly altering the size of standoffs and positioning pulleys in slightly altered locations on the shaft.

The final and greatest issue which the team ran into involved the gearboxes. Upon choosing to use Onvio AP-020 gearboxes, the team was presented with a CAD model of the proposed gearbox. Upon receiving the gearbox, the team found that the attached motor plate as well as the back half of the gearbox was entirely different from the original gearbox CAD. This caused a serious issue as it would inset the motors an extra half inch into the frame. Given that the motors were designed with 0.125 inches of clearance this presented a serious issue. The issue was solved in two ways. First the team modifying the motor mounts of the commercially developed gearboxes and brought them down from a 12mm profile down to a 6mm profile. This meant we still needed to account for roughly another 6mm of extended gearbox length. The solution was to use the existing CAD model of the system and shift around the inner components. Specifically, the team made larger inserts for the electronic inserts and redesigned the battery box to use thinner foam. Using these two solutions the interferences between the motors and the inner body were solved. Looking back at the design requirements the shoulder and knee system driven by a cable transmission could now be considered completed and functioning.

4.0.7 Ankle Assembly

The ankle system assembly proved to be a fairly simple system to assembly. The assembly process was broken up into two steps, the first of which being the assembly of the foot off the robot and then the integration between the foot, lower leg and ankle linkage. With these distinct steps in place any assembly issues could be quickly identified and solved. The first assembly issue came when attempting to assemble the feet plates together and have the support shafts interface properly with all four plates. By creating a proper assembly procedure these fit issues were solved. The proper method for assemble is as follows.

1. Pressfit bushings into lower leg
2. Outer foot pieces screwed together
3. Two halves screwed together
4. Insert support shafts and screw or pin in place
5. Put toes in place and insert spacing shaft
6. Mount toes to screw supports for foot
7. Attaché spring system clevis to back of foot

The final front foot assembly attached to the lower leg can be seen in Figure 124. With the foot properly assembled, the final linkage installment only required the gas spring to be mounted with the shoulder bold to the leg and the shaft to the foot.

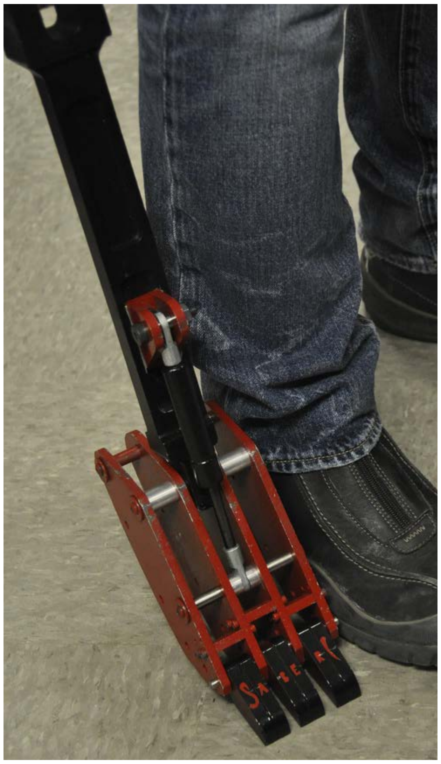


Figure 124 - Fully Assembled Front Ankle

While both the front and rear ankle assemblies require different installation procedures there is a strong similarity with their assembly ideals. The assembly of the rear foot followed the same assembly instruction as the front foot. However, the difference in the rear spring linkage forces the rear foot to be assembled in a slightly more complicated process. A complete rear foot must be attached to a complete rear ankle linkage and then this system can be attached to the upper leg. A full rear linkage system can be seen in Figure 125.

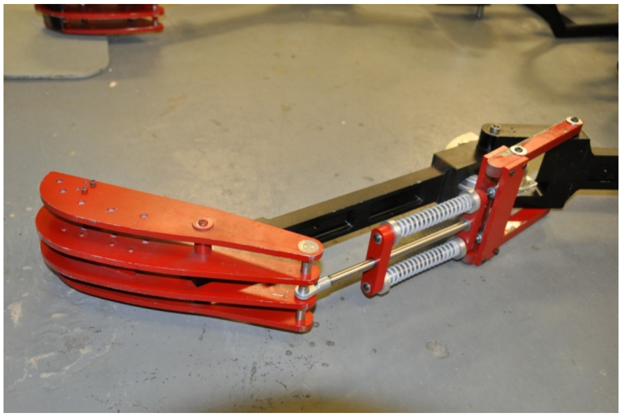


Figure 125 - Fully Assembled Rear Ankle

With a complete leg system (that being front and rear legs) assembled it was starting to become evident that the foot interface with the ground would not provide the necessary grip to support the weight of the robot. Most of the testing surfaces that the robot would be encountering consisted of smooth concrete and it was seen that the small aluminum plates would slip with even a small force applied to them. In an effort to reduce this slippage a steel reinforced rubber substrate was molded to the contour of the bottom of the feet as see in Figure 126.

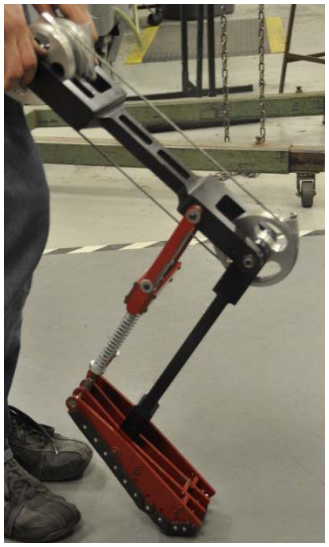


Figure 126 - Rear Ankle With Shoes

4.1 Electrical System

4.1.1 Robot Wiring

Wiring the robot proved to be non-trivial. While the team was able to use the wiring diagram as an initial stepping point, logistics still need to be sorted through to determine the most ideal way to distribute power and signal through the entire system. The team deviated from the original wiring diagram in a few small ways. The first deviation dealt with the placement of the Maxon motor controllers. Due to the multiple iterations of motor controllers the team found that there was an extremely limited amount of space for the motor controllers inside the frame. Seeing no other viable options, the choice was made to place the Maxon motor controllers on the shoulder plates. This was clearly not an ideal solution since the controllers would constantly move relative to the motors themselves. The motors would need to connect to the controllers to pass through encoder data as well as motor power. Clearly, this was not an ideal solution. Upon wiring the robot, the team was able to find a more ideal solution for the controller location. The team would simply place the controllers directly underneath the Maxon motors. A lasercut piece of acrylic was attached with

Velcro to secure the controllers down. This led to a much improved solution since both the motors and controllers were not moving relative to one another.

The next issue was figuring out how to run both power and signal between the two halves of the frame. The team first investigated the use of IGUS chain as well as various nylon mesh materials. All of the materials suffered from the same flaw. While they were able to take up slack in one direction, they would suffer from lack of flexibility in another. The body joint is able to compress in two distinct directions and finding a method to transfer cables safely and securely over both directions proved to be exceedingly difficult. Finally, the team was able to settle on an appropriate carrier method in the form of highly flexible vacuum hose. This material could both compress with ease and bend in a secondary direction. Additionally the material was also very easy to secure since it resembled a large flexible thread. The team used four short sections of this vacuum hose to transfer power and signal on opposite sides of the frame. The team followed the original plan and ran the power cables on the top portion of the robot while running all signal wires at the bottom portion. A picture of the two vacuum tubes implemented in the system can be seen in

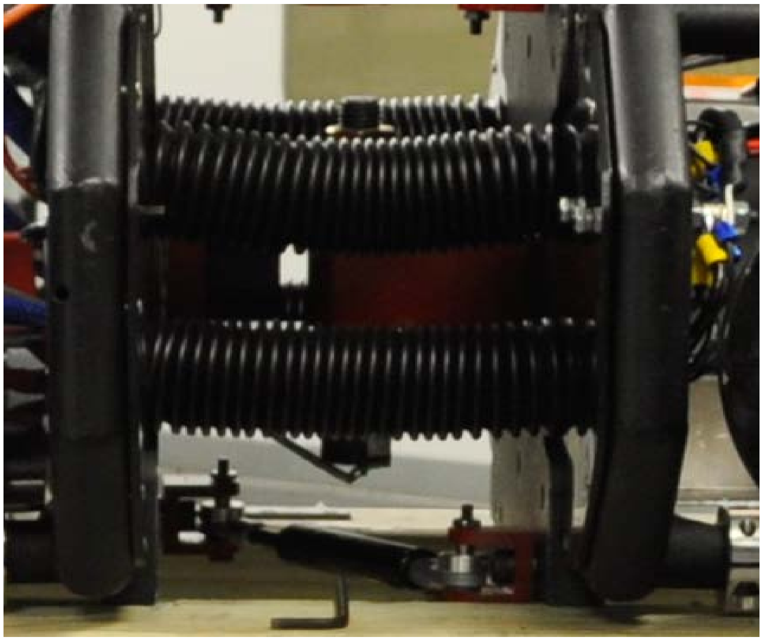


Figure 127 - Vacuum Hoses Separating the Two Halves of the Robot

The next challenge came in the form of making each leg modular. During testing phases the robot's legs would consistently come on and off the robot frame. However, the legs contained 4 potentiometers each containing three wires for power, signal, and ground. This would mean a total of twelve wires which would consistently need to be rewired into the system. The team was able to find an elegant solution by using a DB-15 connector cable at the frame of the robot and a matching connector attaching to the leg potentiometers.

The final deviation from the initial wiring plan was the method of triggering the emergency stops of the system. As planned, the team implemented two emergency stops on each half of the robot. While the original plan had been to use a relay bank to trigger an immediate halt in each motor controller, the team abandoned this option due to both the size of the relays as well as the general complexity. Instead, the team was able to find a much simpler system in the form of a single large contactor. This contactor would lay in line with the two emergency stops. If either stop was triggered then the normally open contactor would switch position and stop all operation of the motors.

As a whole, the overall wiring of the system turned out very well. 10 gauge wire was used on all of the batteries, 14 gauge for all the motors and 24 gauge for all of the signal wires. It should be noted that the team took great care to cover all wires in meshing to ensure that no wires would be torn through due to extended wear. The team also effectively utilized a ground and power bus to distribute all of the source power to the motors and electronics. Finally, the team made sure to check for any interference in the system and make sure that the frame was isolated from both the ground and power bus.

4.1.2 Motion Control

Controlling the motion of the motors to pass through a gait consisted of controlling two independent motors per leg. The initial tests were performed on a prototyping board without any load (Figure 128). The latter tests contained the load of the robot's leg, weighing approximately 50 pounds (Figure 129).

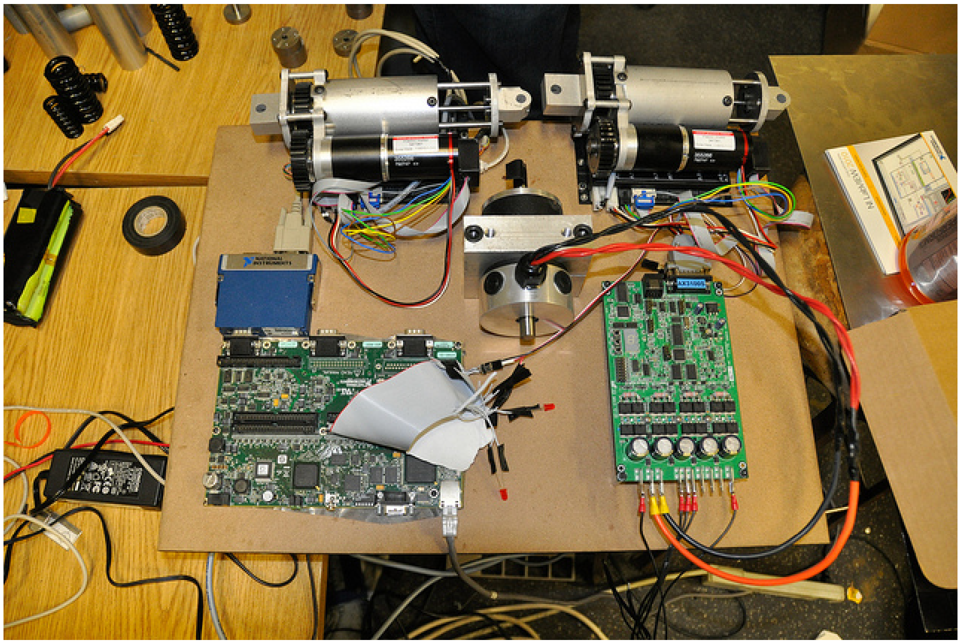


Figure 128 - Testing motor control without load

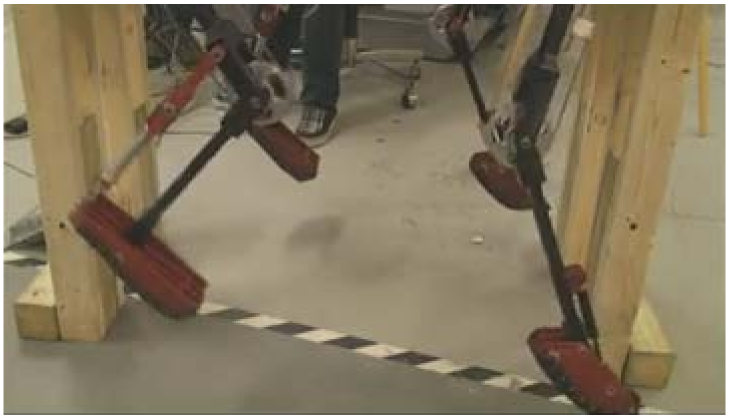


Figure 129 - Testing full motion control with full body load

The initial tests were developed to test the ability to communicate and control independent electronic components. Because of the high level of hardware in the system, in order to test motor controllers properly there must be a large architecture set up. To move a shoulder or elbow, data must be passed from the EeePC in a C++ program to the sbRIO using G code through UDP. The G code would manipulate the data into a PWM signal and send it to the proper motor controller. To move a hip, instead of a PWM signal, a CAN frame is sent through the CAN Module attached to the sbRIO to an EPOS2 node. This high degree of data manipulation, though very thorough and powerful, is very difficult to initially setup. These tests were a way to prove the conceptual system architecture would succeed.

Once the system was built a more thorough test of sending UDP packets was then instantiated, the concept of which was to send a position regarding where the robot should be in the next 50ms (hence why we calculated our gaits with respect to time). As these packets were sent the data was then parsed on the FPGA and interpolated to 2ms duty cycle streams into the motor controllers. This functionality, too, was a success. Initially setting the P value in PID control to 1 made oscillation too high, but tuning this value down to .3-.5 range for a given leg stopped oscillations and allowed for a more fluid motion. The system was able to have all four legs successfully go through a walking gait.

4.2 Software System

4.2.1 Simulation

Through the MATLAB simulation, the team was able to determine that it was indeed just a visual simulation that simplified a lot of our modeling. The graphical user interface mixed with the high level programming of MATLAB has poor computational runtime. Rewriting this code in C++

due to modulatory with the software architecture was chosen to reduce the gait generation run time. The simulation's ability to derive proper mathematical modeling of the entire system was accurate and successful, as was shown in the test in Figure 129.

The DH parameters used in Table 9 modeled the system accurately, neglecting the central body joint. Gait generation was computed at a rate of ~1KHz with C++, as opposed to ~1Hz with MATLAB. The logged gait generation data, as shown in Figure 130, shows the large capabilities of data logging for future development and research. This allowed for significant improvements in the increase of resolution throughout the dataset.

Using the forward kinematics was important to calculating direct positions of a given rigid body in the simulation. When implementing the computational engine for gait calculations for the actual robotic platform and using C++, forward kinematics were unnecessary.

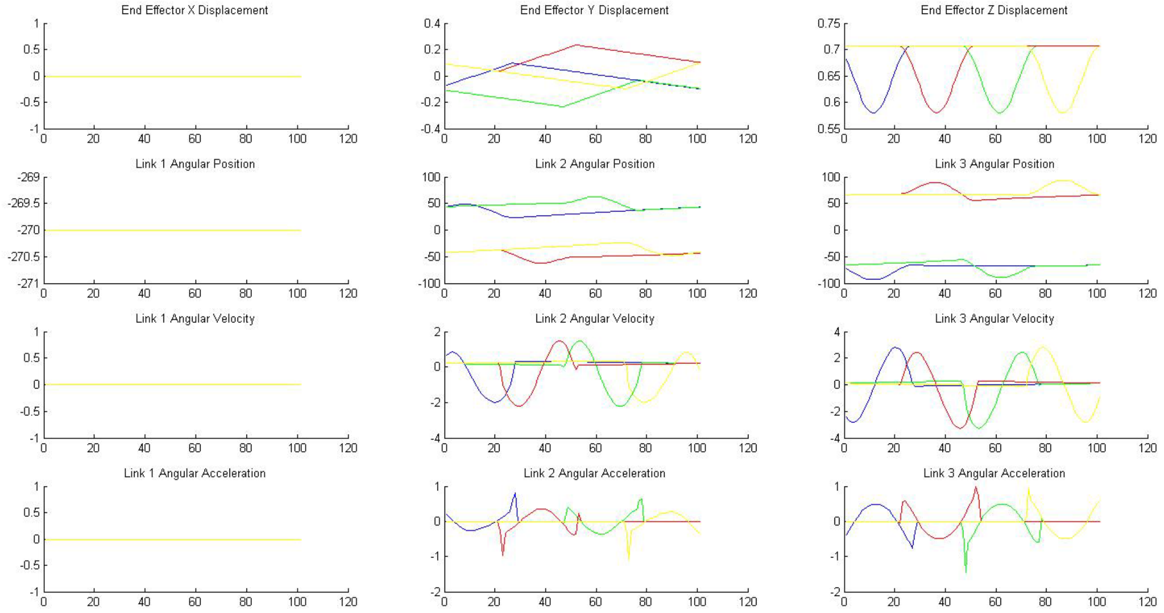


Figure 130: C++ Data Acquisition of a forward gait.

The initial torque estimates were derived assuming with 3 legs on the ground the down force would give the robot a weight of approximately 240 pounds. This value was 30 pounds off, and compensation for this was predicted in motor selection.

The modeling of the robot was considered a success in the accuracy of predictions of inverse kinematics and joint angles required for locomotion. The gait analysis performed on the system was considered a success as well in the implementation of the robot. Stability and the central body joint were neglected in the simulation. The torque analysis from the simulation has been tested on a small scaled, currently testing of this has included moving a leg to desired positions and has been success. This has been using a fraction of the power capable because the robot has not stood yet without support. An unsuccessful test was performed by attempting to stand the robot on three legs. Because of a mechanical failure regarding the cables, the test was never able to be completed.

4.2.2 Mapping and Path Planning

The three-dimensional map constructed from the LIDAR proved to be an accurate method to gain information of the environment. The LIDAR used (Figure 20) is a high resolution LIDAR that provides fine distance data. The filter applied worked well to reduce erroneous data points, as shown below:

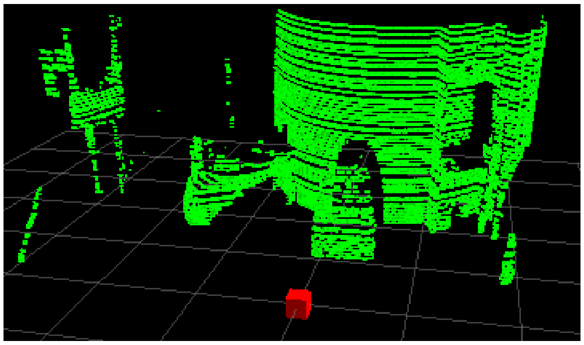


Figure 131 - Map Pre-filtering

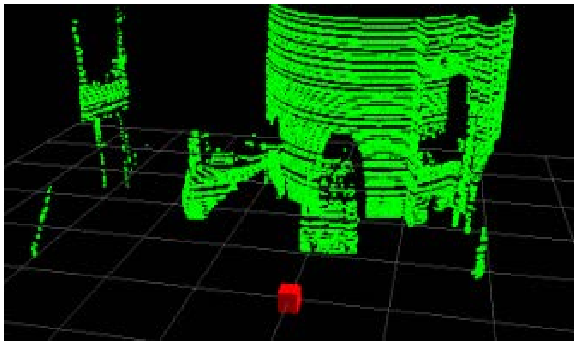


Figure 132 - Map Post-filtering

Once the noise of the map was reduced, the occupancy grid (Figure 108) was visually compared with the top-down view of the three-dimensional map (Figure 107). This visual analysis was performed on several different LIDAR scans to verify that the occupancy grid was an accurate representation of the world. For each map generated, the algorithm for constructing the occupancy grid proved to be correct, just as in the aforementioned figures.

The generated C-space was then validated by displaying a map with the expanded occupancy grid (shown below). As can be seen, the obstacles from Figure 108 were appropriately expanded to account for half of the width of the robot. Debug statements in the robot reported each square was expanding by 6 squares – this corresponded to 60cm, which accounts for half the width of the robot (the full width is about one meter) plus a 10cm tolerance.

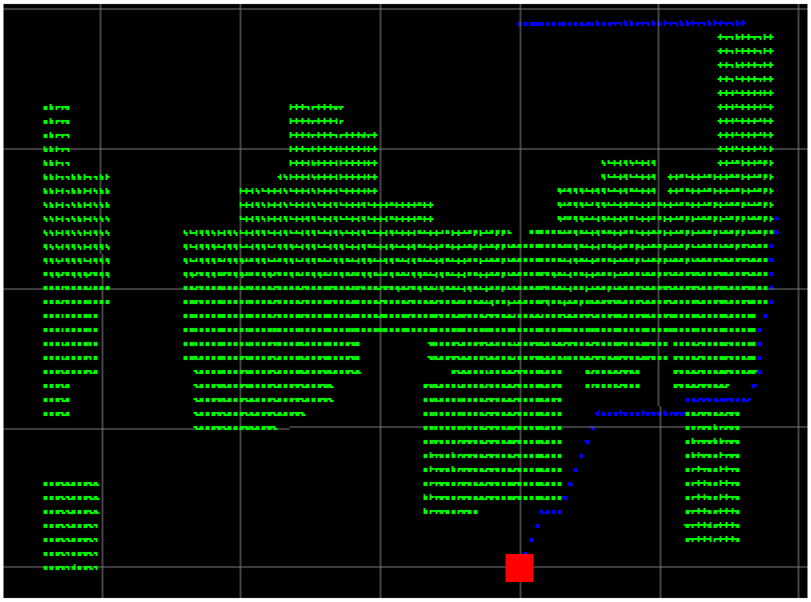


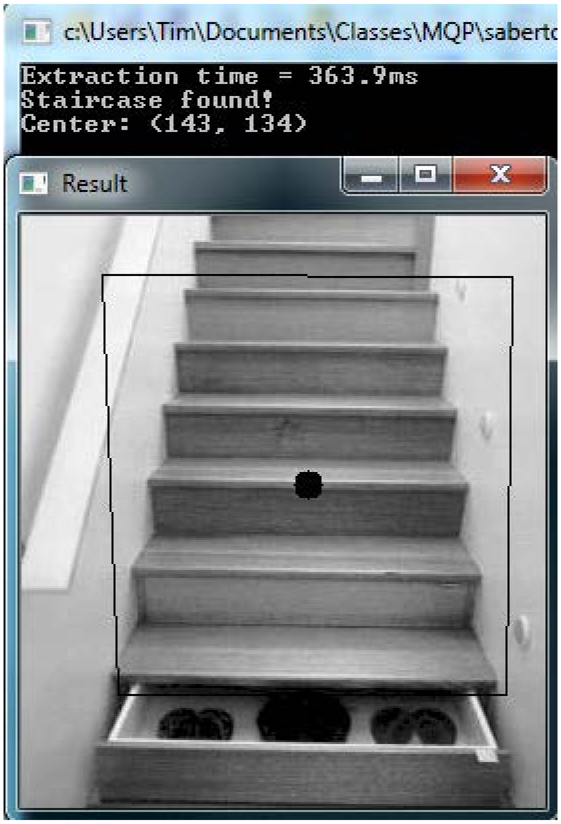
Figure 133 - Occupancy Grid With C-Spacing

Next, path planning was visually verified. The A* algorithm was judged to be complete after testing the algorithm on a series of large maps with different starting and ending nodes. The heuristic used to estimate the goal (the straight line distance to the end goal) is known to never overestimate, therefore it is admissible and the algorithm is optimal. When viewing Figure 109, the path appears to be suboptimal. However, this is simply because that figure does not show the nodes expanding via C-Spacing. When viewing Figure 133, it is clear that the path is optimal when the expanded nodes are taken into consideration.

These results satisfy the requirements of using the LIDAR for obstacle avoidance and demonstrating A* for path planning, and the requirement of calculating distances to objects.

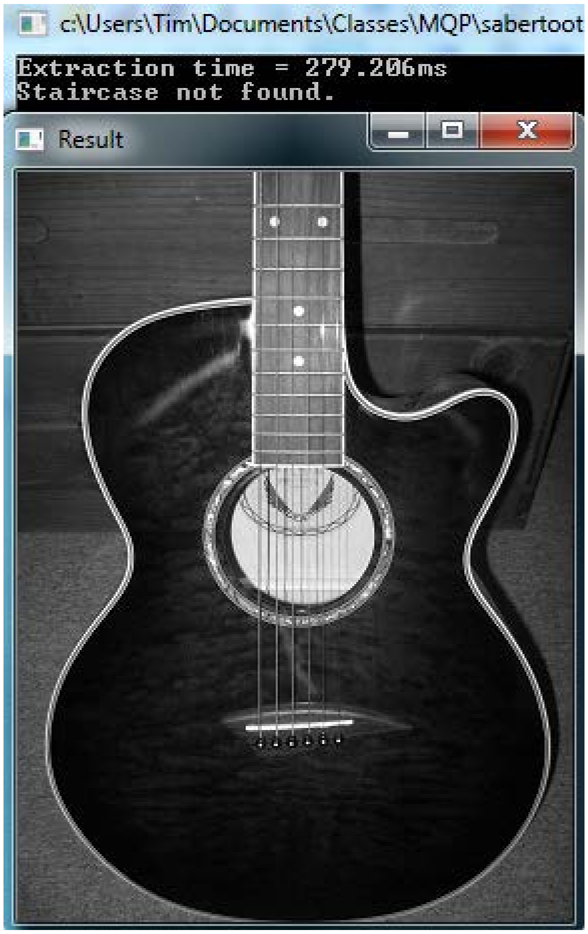
4.2.3 Staircase Recognition

The camera used for a PlayStation 3 Eye, due to its low cost for a higher resolution image. The team used OpenCV's image software library for the SURF framework provided by the library. The algorithm was modified to record the center position of the staircase, if one was found, and the time it took to extract the features and probabilistically determine if it is a staircase. The algorithm stores a database of key features found in images used to "train" the algorithm. Key features are then extracted from a test image, and these features are compared with the database features. The test image is provided by the camera – a live video feed is streamed from the camera, and snapshots are taken to use as a test image. If enough clusters of features are found to be similar, then the image is determined to contain a staircase. Below are images of a positive test and a negative test of the SURF implementation:



```
Extraction time = 363.9ms
Staircase found!
Center: (143, 134)
```

Figure 134 - Positive Test of Staircase Recognition



```
Extraction time = 279.206ms
Staircase not found.
```

Figure 135 - Negative Test of Staircase Recognition

This correct identification satisfies the requirement of recognizing staircases with the on-board camera. However, the detection system was not incorporated on the robot in a real-time environment.

4.2.4 IMU Stabilization

The inertial measurement unit used by the robot was the MicroStrain 3DM-GX3-25 sensor. The team developed a Linux software driver to interface with the IMU. The data received from the IMU is in the form of IEEE floating point hexadecimal as a string, so an algorithm was written to convert this string to a more useful decimal double. From this, the orientation, acceleration, change in velocity, and other data could be read and recorded from the sensor. From this data (namely the orientation data), the roll, pitch, and yaw of the robot can be calculated. The IMU was placed on the body joint of the robot. From there, the orientation of any point on the robot can be calculated via a series (or single) transformation and rotation matrices.

While the ability to record the acceleration, velocity, and orientation of the robot was successfully achieved, that data has not been implemented in a well-tested stability algorithm. The project has not yet reached that point.

4.2.5 Communication Protocol

The communication protocol, described in Section 3.5.5, was successfully implemented in the system. As described, UDP communication was used to send the packets between the Eee PC and SBRIO. During testing, we tested the main weakness of UDP – that it is connectionless and does not use handshaking, so packet transmission is not guaranteed. However, since the packets were only being sent across a few feet of Ethernet cable directly connected on both ends, the team found that not a single packet was lost during transmission. However, even if a packet was lost, it would not negatively affect the system because the next packet is sent 50 milliseconds later.

The system was tested fully using the software architecture described in Section 3.3. The robot was mounted on a testing table where the legs had no contact with the ground. From there, the executive was used to instruct the planner to plan a path walking straight ahead two meters (without using data a three-dimensional map, as this was not needed – instead, an empty occupancy grid was used). The planner was then instructed to use that path to calculate gaits for the robot. The pilot, where the UDP communication is handled, then successfully sent the potentiometer positions to the SBRIO. This was verified by monitoring the packets in Labview as they came into the

SBRIO. The SBRIO then successfully sent the data to the motor controllers, which was verified by monitoring the legs move through the predetermined gaits.

This successful communication satisfies the requirement of implementing a network communication protocol for communication between all on- and off-board processors.

4.3 Final Design Overview

As a final inspection, the team went back and reviewed their initial task specifications. The team compiled the specifications into Table 21and determined which task specifications had and hadn't been met. While there were a few untested specifications overall the team considers a project a success in terms of meeting the desired objectives.

Design Specification	Completed
Develop a robust mechanical base for legged locomotion research	Full Mechanical System Developed and Tested
Robot must weigh less than 200 pounds	Final Robot weight 270
Must be capable of carrying a 30 pound payload the size of a cinder block	Untested
Dynamically imitate animal walking, trotting and bounding gaits	Simulation and Controls developed, Untested
Mechanically designed to be physically capable of walking, trotting and bounding gaits	Completed
Develop compliant spring system allowing the robot to reproduce a SLIP model	Completed
Robot must be untethered	Completed
Recover and monitor energy lost with each step	Completed
Achieve a statically stable 5ft/sec walking gait	Untested
Develop a mechanically complaint ankle system able to recreate the "toe-to-heel" movement of quadrupedal animals	Completed
Each leg must contain three degrees of freedom called the "hip", "shoulder", and "knee"	Completed
Develop an flexible midsection of the robot that has 2 degrees of freedom.	Completed
Develop software to learn the robot's surroundings using: IMU for robot stability	Untested
Develop software to learn the robot's surroundings using: LIDAR for obstacle avoidance	Completed
Develop software to learn the robot's surroundings using: Camera for staircase recognition	Completed
Demonstrate employment of A* search algorithm for path planning on the robot side	Completed
Must contain a simple network communication protocol to allow for communication between all on- and off-board processors.	Completed

Table 21 - Task Specification Overview

CHAPTER 5: CONCLUSIONS AND RECCOMENDATIONS

This chapter provides concluding remarks on the research and analysis conducted by the team. The accompanying recommendations are based on these conclusions and the initial set of project goals and are provided as future goals for the project.

5.0 Conclusion

Before the design of the mechanical systems was conducted, significant research was conducted on the different aspects of an animal body. This was done because one goal behind the project was to mimic the gait of lion-like animals. Specific research was conducted on animal locomotion and the biological functionality of the body, legs, and ankles of lions. This research provided the necessary background to design a mechanically compliant leg system for the robot. Further background research was conducted on individual components of the software system, including path planning and image recognition. This was to provide sufficient information for the team to decide what algorithms that would work best with respect to the different components.

After the initial background research was conducted, the next step was to develop a mathematical and mechanical simulation as a proof of concept for the project. The model also served to provide data about the torque exerted on the link joints, which was needed to come to a reasonable conclusion as to what motors were needed to run the system. The model was also used to verify the six types of gaits (Figure 14) the team decided to implement. At each time step in the simulation, the exact position of each joint is also plotted to verify the inverse kinematics of the system computed by the simulation. Furthermore, the simulation was useful to model the robot moving at different speeds and stride lengths, allowing the team to confidently ensure that balance at all times was maintained.

Simultaneous to the development of the simulation, the team designed each component of the system in SolidWorks 2010. Careful attention was paid to the mechanical compliance aspect of the robot – a large influence over the robot design was the ability for the robot to be mechanically compliant with uneven terrain. After an initial design was complete, all crucial subsystems were prototyped and tested for functionality. All flaws in the design were noted and fixed in the SolidWorks model. Once the mechanical design was complete, manufacturing and assembly began. The majority of the robot was manufactured in-house using Worcester Polytechnic Institute's CNC machines. A small portion of the robot was outsourced to be waterjetted.

Software development began shortly after the simulation was complete. First, development began on the object recognition software. The two algorithms researched (SIFT and SURF) were discussed, and the team decided to implement the SURF algorithm due to its ability to handle angled and skewed input data. Next, development began on the LIDAR vision system, including a driver for the LIDAR, software to build a three-dimensional map, software to construct the two-dimensional occupancy grid, and software to use that grid for path planning. Software was then developed to interface with the IMU. Once these components were created and tested, construction of the software architecture (Figure 103) and user interface (Figure 104) began. The main purpose of the architecture was to provide an abstraction for each part of the software system, with an organized manager to run oversee the system. The SBRIO software was developed in LabVIEW, and each virtual interface of the LabVIEW system was tested individual with the motor controllers.

Testing during development of all aspects of the project was a crucial aspect to ensure the correctness of the design of each system. The tests completed (e.g. prototypes, simulations, and running the software) pointed out the major flaws in the designs, allowing the team to correct the flaws before moving forward. The final test completed was the entire system interacting together with the robot mounted on a test harness, with the software instructing the controllers of the robot to successfully walk through gaits (described in Section 4.2.5).

5.1 Mechanical Recommendations

To improve the mechanical aspects of the robot, the team recommends exploring the following systems in more depth.

- Explore mechanics of simulation
- Explore spring system in more depth, gather data from it

While the team developed a very in depth simulation there was much more which could be done. The team was able to develop a simulation which simulated a floating body and contained contact forces, however, the contact forces were simply a pass fail system and did not take into account the time it would take for the foot to come in full contact with the floor. Simulating this contact force over time would give the team a better understanding of how the robot is interacting with the overall environment. Additionally, while the team began preliminary stabilization calculations, expanding them would also allow for better understanding of the full system. The team modeled the center of gravity of each individual link as well as the center of gravity of the body. Knowing the effective radii as well as the overall weight of each individual component the team could calculate the center of gravity of the robot. Next, knowing how many feet are on the ground we can also actively calculate the polygon of contact of the robot and ensure that the robot's center of gravity falls within the polygon. Modeling more components would give the team a much more accurate reading on the location of the center of gravity. Finally, in the current MATLAB simulation the springs are not modeled. In future versions the team recommends looking more in depth at the spring system and modeling the compression of the springs and actively calculating how much energy is theoretically recovered from the spring system.

The team also recommends studying the spring system in greater depth. While the team was able to develop and implement the SLIP model into the system there was no time to test it in full depth or test to see how much energy the springs were able to recover. Additionally, the team was unable to test for optimal spring stiffness for our specific system. Simply put, the team was able to implement exactly one form of spring and complete only very limited testing. A full detailed analysis of the spring system could help in identifying the need for well-tuned spring stiffness and could help future generations better understand quadrupedal and bipedal locomotion.

5.2 Software Recommendations

To improve the software system of the robot, the team recommends the following:

- Explore adaptive gaits
- Implement dynamic balancing
- Integrate vision systems
- Add monitor block to architecture

Currently, the robot has successfully moved through only one of the six desired gaits (the lateral walking gait). Further testing should be completed to ensure the robot is capable of safely traversing through the other gaits. For stabilization purposes, some of the gaits require the robot to move at a higher speed than the gait currently used because those gaits require more than two legs off the ground at once. Furthermore, the IMU should be further developed to incorporate dynamic balancing. The driver and data acquisition software has been written, but it needs to be interfaced with the motion of the robot to correct for off-balance motion. Next, the vision system should be integrated with the robot. The vision system has been entirely developed and tested, as well as interfaced with the software architecture. However, that feature is not currently utilized by the robot because the testing phase has not yet gotten to the point where the robot is capable of walking completely autonomously. Finally, a monitor block should be added to the software architecture, as in Figure 136. The purpose of the monitor will be to intercept incoming UDP packets from the SBRIO. Currently, the data coming in from the sensor block is not utilized – instead, it is used for logging purposes. The monitor will serve to validate the information coming in, and determine if possible errors occur, such as corrupt data and unreasonable data (if a value coming in is significantly different than the previous value). The monitor can also perform position error calculations; currently, this is only performed on the SBRIO and the software architecture does not know these errors.

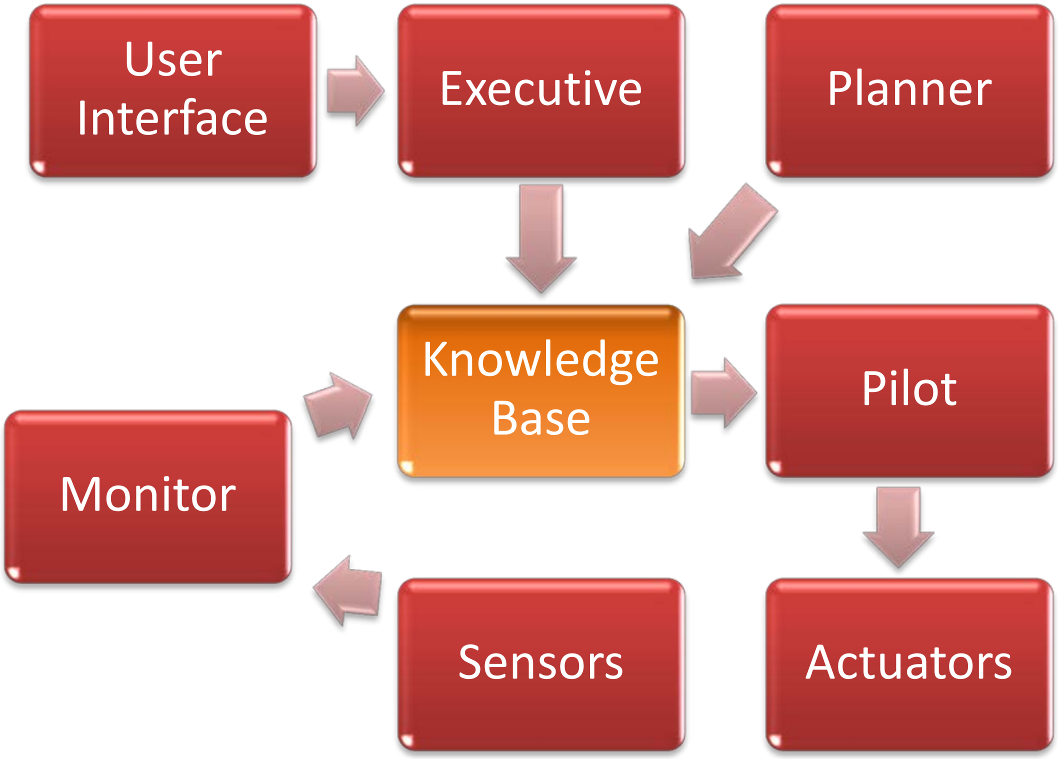


Figure 136 - Software architecture with monitor

WORKS CITED

Adamczyk, Peter, Steven Collins, and Arthur Kuo. "The Advantages of a Rolling Foot in Human Walking." *Journal of Experimental Biology* 209 (2006): 3953-963. Print.

Bauer, Oliver Francis. *Digital Control of Quadruped Robot. Autonomous Systems Lab.* Swiss Federal Insititute of Technology Zurich, Spring 2009. Web.

Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. *SURF: Speeded Up Robust Features. ETH Zurich.* Katholieke UNiversiteit Leuven. Web.

Berkemeier, M. D. "Modeling the Dynamics of Quadrupedal Running." *The International Journal of Robotics Research* 17.9 (1998): 971-85. Print.

Boston Dynamics. *BigDog Overview.* Boston, 2008. Print.

Bradski, Gary R., and Adrian Kaehler. *Learning OpenCV: [computer Vision with the OpenCV Library].* Sebastopol: O'Reilly, 2008. Print.

Carrier, David, Colin Gregersen, and Natalie Silverton. "Dynamic Gearing in Running Dogs." *Journal of Experimental Biology* 201 (1998): 3185-195. Print.

Chen, Wenjie, K. H. Low, and S. H. Yeo. "Adaptive Gait Planning for Multilegged Robots with an Adjustment." *Cambridge University Press* 17 (1999): 391-403. Print.

"Compression Springs | Stock Springs | Lee Spring." *Lee Spring - Precision Stock & Custom Springs.* Web. 26 Apr. 2011. <http://www.leespring.com/int_learn_compression.asp>.

Dechter, Rina, and Judea Pearl. "Generalized Best-first Search Strategies and Optimality of A*." *Journal of the ACM* 32.3 (1985). Print.

Dijkstra, Edsger W. "A Note on Two Problems in Connexion with Graphs." *Numerische Mathematik* 1 (1959): 269-71. Print.

Farley, Claire, James Glasheen, and Thomas McMahon. "Running Springs: Speed and Animal Size." *Journal of Experimental Biology* 185 (1993): 71-86. Print.

Fu, Chenglong, and Ken Chen. "Gait Synthesis and Sensory Control of Stair Climbing for a Humanoid Robot." *IEEE Transactions on Industrial Electronics* 55.5 (2008): 2111-120. Print.

Fukuoka, Y., H. Kimura, and A. H. Cohen. "Adaptive Dynamic Walking of a Quadruped Robot on Irregular Terrain Based on Biological Concepts." *The International Journal of Robotics Research* 22.3-4 (2003): 187-202. Print.

Goslow, G. E., H. J. Seeherman, C. R. Taylor, M. N. McCutchin, and N. C. Heglund. "Electrical Activity and Relative Length Changes of Dog Limb Muscles as a Function of Speed and Gait." *Journal of Experimental Biology* 94 (1981): 15-42. Print.

Gray, J. *Croonian Lecture: Aspects of Animal Locomotion*. The Royal Society, Winter 1939. Web.

Gregersen, Colin S., and David R. Carrier. "Gear Ratios at the Limb Joints of Jumping Dogs." *Journal of Biomechanics* 37 (2004): 1011-018. Print.

"GSD Hindquarters, Hindquarters in German Shepherds, German Shepherd Dog Hindquarter." *About German Shepherds, about German Shepherd, German Shepherd Dogs, German Shepherd Information, German Shepherds Info*. Web. 26 Apr. 2011. <http://www.aboutgermanshepherddog.com/page.php?page_id=109>.

H., Walter. "Normal Joint Range of Motion in the Dog and Cat." *CAL Home*. Web. 26 Apr. 2011. <http://cal.vet.upenn.edu/projects/saortho/appendix_b/appb.htm>.

Herbert, Sam, Andrew Drenner, and Nikolas Papanikolopoulos. "Loper: A Quadruped-Hybrid Stair Climbing Robot." *IEEE International Conference on Robotics and Automation* (2008). Print.

Herr, H. M. "A Galloping Horse Model." *The International Journal of Robotics Research* 20.1 (2001): 26-37. Print.

HIldebrand, M. "Symmetrical Gaits of Horses." *Science* 150 (1965): 701-08. Print.

Holmes, Philip, Robert Full, Dan Koditschek, and John Guckenheimer. "The Dynamics of Legged Locomotion: Models, Analyses and Challenges." *SIAM Review* 48.2 (2006): 207-304. Print.

"Interaction between Trajectory Planners for Little Dog." *SCHOOL OF COMPUTER SCIENCE, Carnegie Mellon*. Web. 26 Apr. 2011. <http://www.cs.cmu.edu/~mmakatch/courses/kdc/project/>.

Ivanenko, Y. P., Y. S. Levik, V. L. Talis, and V. S. Gurfinkel. "Human Equilibrium on Unstable Support: the Importance of Feet-support Interaction." *Neuroscience Letters* 235 (1997): 109-12. Print.

Kabasawa, Mitsutaka, Satoshi Kagami, Massayuki Inaba, and Hirochika Inoue. *Trapezoidal Gait on Stairs for a Quadruped Robot*. University of Tokyo, 1999. Web.

Kimura, H., Y. Fukuoka, and A. H. Cohen. "Adaptive Dynamic Walking of a Quadruped Robot on Natural Ground Based on Biological Concepts." *The International Journal of Robotics Research* 26.5 (2007): 475-90. Print.

Kolter, Zico J., Mike P. Rodgers, and Andrew Y. Ng. *A Control Architecture for Quadruped Locomotion Over Rough Terrain*. *Computer Science Department*. Stanford University, Fall 2008. Web.

Kuboyama, Kiyokazu, PEng Chen, Hiroyuki Mori, and Toshio Toyota. *Control Method of Walking and Obstacle Avoidance for Quadruped Robot by Classifier System and Genetic Algorithms*. Kyushu Institue of Technology. Web.

Lowe, Davide G. *Distinctive Image Features from Scale-Invariant Keypoints. Computer Science Department*. University of British Columbia, 5 Jan. 2004. Web.

McGuigan, M. P., and A. M. Wilson. "The Effect of Gait and Digital Flexor Muscle Activation on Limb Compliance in the Forelimb of the Horse Equus Caballus." *Journal of Experimental Biology* 206 (2003): 1325-336. Print.

"Morphological Comparison of Big Pantherines and Brown Bears." *Animal vs Animal Forums*. Web. 26 Apr. 2011. <http://animalsversesanimals.yuku.com/topic/2162/t/Morphological-comparison--big-pantherines--brown-bears--Pete.html>.

Nestinger, Stephen S. *Kinematics of Serial Robots*. Worcester: Worcester Polytechnic Institute, 2011. Print.

Nichol, Gordon J., and Kenneth J. Waldron. *Biomimetic Leg Design for Untethered Quadruped Gallop. Mechanical Engineering Department*. Stanford University. Web.

Nichol, J. G. "System Design of a Quadrupedal Galloping Machine." *The International Journal of Robotics Research* 23.10-11 (2004): 1013-027. Print.

Norton, R. L. *Machine Design: an Integrated Approach*. Upper Saddle River, NJ: Prentice Hall, 2005. Print.

Norton, Robert L. Design of Machinery: an Introduction to the Synthesis and Analysis of Mechanisms and Machines. Maidenhead: McGraw-Hill Education, 2003. Print.

Nunamaker, David M., and Peter D. Blauner. "Normal and Abnormal Gait." *CAL Home*. Web. 26 Apr. 2011. <http://cal.vet.upenn.edu/projects/saortho/chapter_91/91mast.htm>.

Olson, John M., and Richard L. Marsh. "Activation Patterns and Length Changes in Hindlimb Muscles of the Bullfrog Rana Catesbeiana." *Journal of Experimental Biology* 207 (1998): 2763-777. Print.

"Oricom Technologies Leg Mechanics." *Oricom Technologies Home Page*. Web. 26 Apr. 2011. <http://www.oricomtech.com/projects/leg-mech.htm>.

Palmer, Luther R., and David E. Orin. "Intelligent Control of High-Speed Turning in a Qudruped." *Journal of Intellegent Robotic Systems* 58 (2010): 47-68. Print.

Pasi, B. M., and D. R. Carrier. "Functional Trade-offs in the Limb Muscles of Dogs Selected for Running vs. Fighting." *Journal of Experimental Biology* 16 (2003): 324-32. Print.

"Physical Structure of German Shepherd Dog: GSD Standard." *German Shepherd Dog Information*. Web. 26 Apr. 2011. <http://aringsburgkennel.blogspot.com/2009/01/physical-structure-of-german-shepherd.html>.

Prilutsky, Bori I., and Vladimir M. Zatsiorsky. "Tendon Action of Two-joint Muscles: Transfer of Mechanical Energy between Joints during Jumping, Landing and Running." *Journal of Biomechanics* 27.1 (1994): 25-34. Print.

Prilutsky, Boris I., W. Herzog, and T. Leonard. "Transfer of Mechanical Engergy between Ankle and Knee Joints by Gastrocnemius and Plantaris Muscles during Cat Locomotion." *Journal of Biomechanics* 29.4 (1996): 391-403. Print.

QiDi, Wu, Liu ChengJu, Zhang JiaQi, and Chen QiJun. "Survey of Locomotion Control of Legged Robots Inspired by Biological Concept." *Science in China Series F* (2009). Print.

Raibert, Marc. "Legged Robots." *Communications of ACM* 29.6 (1986): 499-517. Print.

Rummel, Juergen, and Andre Seyfarth. *Stable Runing with Segmented Legs. Locomotion Labortatory*. University of Jena. Web.

Rutishauser, Simon. *Cheetah - Compliant Quadruped Robot. Biologically Inspired Robotics Group*. Ecole Poltechnique Federale De Lausanne, 11 Jan. 2008. Web.

Schmiedeler, J. P. "The Mechanics of Quadrupedal Galloping and the Future of Legged Vehicles." *The International Journal of Robotics Research* 18.12 (1999): 1224-234. Print.

Schmittwilken, Jorg, and Lutz Plumer. *Model Selction for Compositie Objects with Attribute Grammars. Institute of Geodesy and Geoinformation*. University of Bonn, Summer 2009. Web.

"SiriusDog.com - Anatomy of the German Shepherd." *SiriusDog.com -*. Web. 26 Apr. 2011. <http://www.siriusdog.com/anatomy-german-shepherd-dog.htm>.

Sreenath, Koushil, Hae-Won Park, Ioannis Poulakakis, and J. W. Grizzle. *A Compliant Hybrid Zero Dynamics Controller for Stable, Efficient and Fast Bipedall Walking on MABEL*. University of Michigan, Summer 2009. Web.

Stentz, Anthony. *Optimal and Efficient Path Planning for Unknown and Dynamic Environments. The Robotics Institute*. Carnegie Mellon University, Aug. 1993. Web.

Szeliski, Richard. *Computer Vision: Algorithms and Applications*. Rep. Print.

Talebi, S., M. Beuhler, and E. Papadopoulus. *Towards Dynamic Step Climbing for a Quadruped Robot with Compliant Legs. Ambulatory Robotics Laboratory*. McGill University. Web.

Thunder Power RC. *THPS Safety Warning*. Las Vegas: Thunder Power RC, 2009. Print.

Todd, Carl David. *The Potentiometer Handbook: Users' Guide to Cost-effective Applications*. New York: McGraw-Hill, 1975. Print.

Trahey, Steve. "Choosing a Code Wheel: A Detailed Look at How Encoders Work." *Journal of Modular Embedded Design* (2008). Print.

Tuttle, Timothy D. *Understanding and Modeling the Behavior of a Harmonic Drive Gear Transmission. Mechanical Engineering Department*. Massachusetts Institute of Technology, 1992. Web.

Usherwood, James R., Sarah B. Williams, and Alan M. Wilson. "Mechanics of Dog Walking Compared with a Passive, Stiff-limbed, 4-bar Linkage Model, and Their Collisional Implications." *Journal of Experimental Biology* 210 (2007): 533-40. Print.

Usherwood, James R., Sarah B. Williams, and Alan M. Wilson. "Mechanics of Dog Walking Compared with a Passive, Stiff-limbed, 4-bar Linkage Model, and Their Collisional Implications." *Journal of Experimental Biology* 210 (2007): 533-40. Print.

Van, Frank. "Movements of the Dog - Gaits and Locomotion." *Lhasa Apso Breeder, EL Minja's Lhasa Apso Dogs Kennel*. Web. 26 Apr. 2011. <http://www.el-minjas.com/Movements.htm>.

Webster, Robert. "Object Capture with a Camera-Mobile Robot System." *IEEE Robotics & Automation Magazine* Mar. 2006: 85-88. Web.

Whitmore, Frank, and Helen Foster. *Panthera Atrox from Central Alaska. US Geological Survey*. University of Nebraska - Lincoln, Winter 1967. Web.

Zhou, Debao, K. H. Low, and Teresa Zielinska. "An Efficient Foot-force Distribution Algorithm for Quadruped Walking Robots." *Robotica* 18.4 (2000): 403-13. Print.

APPENDIX A: MAPLE CODE OF FORWARD KINEMATICS

Imports

$with(linalg):$

$with(LinearAlgebra):$

Constants

$g := 9.81:$

$mass1 := m1:$

$mass2 := m2:$

$mass3 := m3:$

$mass4 := m4:$

$mass := mass1 + mass2 + mass3 + mass4:$

Coordinate transformations

T01

$a_1 := .15:$

$d_1 := .064:$

$\alpha_1 := 90:$

$\alpha_1 := convert\left(\alpha_1 degrees, radians\right):$

$$T01 := Matrix\big(\big[\big[\cos\big(\theta[1](t)\big), -\sin\big(\theta[1](t)\big)*\cos\big(\alpha[1]\big), \sin\big(\theta[1](t)\big)*\sin\big(\alpha[1]\big), a[1]*\cos\big(\theta[1](t)\big)\big], \big[\sin\big(\theta[1](t)\big), \cos\big(\theta[1](t)\big)*\cos\big(\alpha[1]\big), -\cos\big(\theta[1](t)\big)*\sin\big(\alpha[1]\big), a[1]*\sin\big(\theta[1](t)\big)\big], \big[0, \sin\big(\alpha[1]\big), \cos\big(\alpha[1]\big), d[1]\big], [0, 0, 0, 1]\big]\big)$$

$$\begin{bmatrix} \cos(\theta_1(t)) & 0 & \sin(\theta_1(t)) & 0.15\cos(\theta_1(t)) \\ \sin(\theta_1(t)) & 0 & -\cos(\theta_1(t)) & 0.15\sin(\theta_1(t)) \\ 0 & 1 & 0 & 0.064 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T12

$a_2 := .381 :$

$d_2 := 0 :$

$\alpha_2 := 0 :$

```
T12 := Matrix([[cos(θ[2](t)), -sin(θ[2](t))*cos(α[2]),
    sin(θ[2](t))*sin(α[2]), a[2]*cos(θ[2](t))], [sin(θ[2](t)),
    cos(θ[2](t))*cos(α[2]), -cos(θ[2](t))*sin(α[2]), a[2]
    *sin(θ[2](t))], [0, sin(α[2]), cos(α[2]), d[2]], [0, 0, 0, 1]])
```

$$\begin{bmatrix} \cos(\theta_2(t)) & -\sin(\theta_2(t)) & 0 & 0.381\cos(\theta_2(t)) \\ \sin(\theta_2(t)) & \cos(\theta_2(t)) & 0 & 0.381\sin(\theta_2(t)) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T12.5

$a_{21} := \frac{1}{2} \cdot l_1 :$

$d_2 := 0 :$

$\alpha_2 := 0 :$

```
T125 := Matrix([[cos(θ[2](t)), -sin(θ[2](t))*cos(α[2]),
    sin(θ[2](t))*sin(α[2]), a[21]*cos(θ[2](t))], [sin(θ[2](t)),
    cos(θ[2](t))*cos(α[2]), -cos(θ[2](t))*sin(α[2]), a[21]
    *sin(θ[2](t))], [0, sin(α[2]), cos(α[2]), d[2]], [0, 0, 0, 1]]) :
```

T23

$a_3 \;:= .381 :$

$d_3 := 0 :$

$\alpha_3 := 0 :$

```
T23 := Matrix([[cos(θ[3](t)), −sin(θ[3](t)) * cos(α[3]),
     sin(θ[3](t)) * sin(α[3]), a[3] * cos(θ[3](t))], [sin(θ[3](t)),
     cos(θ[3](t)) * cos(α[3]), −cos(θ[3](t)) * sin(α[3]), a[3]
     * sin(θ[3](t))], [0, sin(α[3]), cos(α[3]), d[3]], [0, 0, 0, 1]])
```

$$\begin{bmatrix} \cos(\theta_3(t)) & -\sin(\theta_3(t)) & 0 & 0.381\cos(\theta_3(t)) \\ \sin(\theta_3(t)) & \cos(\theta_3(t)) & 0 & 0.381\sin(\theta_3(t)) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T23.5

$a_{31} \;:= \frac{1}{2}\, l_2 :$

$d_3 := 0 :$

$\alpha_3 := 0 :$

```
T235 := Matrix([[cos(θ[3](t)), −sin(θ[3](t)) * cos(α[3]),
     sin(θ[3](t)) * sin(α[3]), a[31] * cos(θ[3](t))], [sin(θ[3](t)),
     cos(θ[3](t)) * cos(α[3]), −cos(θ[3](t)) * sin(α[3]), a[31]
     * sin(θ[3](t))], [0, sin(α[3]), cos(α[3]), d[3]], [0, 0, 0, 1]]) :
```

T34

$a_4 \;:= l_3 :$

$d_4 := 0 :$

$\alpha_4 := 0 :$

$$T34 := Matrix([[\cos(\theta[4](t)), -\sin(\theta[4](t)) * \cos(\alpha[4]),$$
$$\sin(\theta[4](t)) * \sin(\alpha[4]), a[4] * \cos(\theta[4](t))], [\sin(\theta[4](t)),$$
$$\cos(\theta[4](t)) * \cos(\alpha[4]), -\cos(\theta[4](t)) * \sin(\alpha[4]), \mathrm{a}[4]$$
$$* \sin(\theta[4](t))], [0, \sin(\alpha[4]), \cos(\alpha[4]), d[4]], [0, 0, 0, 1]]) :$$

T34.5

$$a_{41} := \frac{1}{2}\, l_3 :$$

$$d_4 := 0 :$$

$$\alpha_4 := 0 :$$

$$T345 := Matrix([[\cos(\theta[4](t)), -\sin(\theta[4](t)) * \cos(\alpha[4]),$$
$$\sin(\theta[4](t)) * \sin(\alpha[4]), a[41] * \cos(\theta[4](t))], [\sin(\theta[4](t)),$$
$$\cos(\theta[4](t)) * \cos(\alpha[4]), -\cos(\theta[4](t)) * \sin(\alpha[4]), \mathrm{a}[41]$$
$$* \sin(\theta[4](t))], [0, \sin(\alpha[4]), \cos(\alpha[4]), d[4]], [0, 0, 0, 1]]) :$$

COG-T02

$$C02 := combine(simplify(multiply(T01, T12)))$$

$$\left[\left[\frac{1}{2}\cos\left(\theta_1(t) - \theta_2(t)\right) + \frac{1}{2}\cos\left(\theta_1(t) + \theta_2(t)\right), -\frac{1}{2}\sin\left(\theta_1(t)\right.\right.\right.$$
$$\left.+ \theta_2(t)\right) + \frac{1}{2}\sin\left(\theta_1(t) - \theta_2(t)\right), \sin\left(\theta_1(t)\right), 0.1905\cos\left(\theta_1(t)\right.$$
$$\left.\left.- \theta_2(t)\right) + 0.1905\cos\left(\theta_1(t) + \theta_2(t)\right) + 0.1500\cos\left(\theta_1(t)\right)\right],$$
$$\left[\frac{1}{2}\sin\left(\theta_1(t) + \theta_2(t)\right) + \frac{1}{2}\sin\left(\theta_1(t) - \theta_2(t)\right), -\frac{1}{2}\cos\left(\theta_1(t)\right.\right.$$
$$\left.- \theta_2(t)\right) + \frac{1}{2}\cos\left(\theta_1(t) + \theta_2(t)\right), -\cos\left(\theta_1(t)\right),$$
$$0.1905\sin\left(\theta_1(t) + \theta_2(t)\right) + 0.1905\sin\left(\theta_1(t) - \theta_2(t)\right)$$
$$\left.+ 0.1500\sin\left(\theta_1(t)\right)\right],$$
$$\left[\sin\left(\theta_2(t)\right), \cos\left(\theta_2(t)\right), 0., 0.06400 + 0.3810\sin\left(\theta_2(t)\right)\right],$$
$$\left.\left[0, 0, 0, 1.\right]\right]$$

$$C02d := map(diff, T02, t) :$$

EXTRACTING x y and z

$C02dx := C02d[1, 4] :$

$C02dy := C02d[2, 4] :$

$C02dz := C02d[3, 4] :$

$C02xd1 := diff\,(C02dx, t) :$

$C02yd1 := diff\,(C02dy, t) :$

$C02zd1 := diff\,(C02dz, t) :$

COG-T03

$C03 := combine(simplify(multiply(T01, T12, T23)))$

$$
\begin{aligned}
&\left[\left[\frac{1}{2}\cos\left(\theta_1(t)-\theta_2(t)-\theta_3(t)\right)+\frac{1}{2}\cos\left(\theta_1(t)+\theta_2(t)+\theta_3(t)\right),\right.\right.\\
&\quad -\frac{1}{2}\sin\left(\theta_1(t)+\theta_2(t)+\theta_3(t)\right)+\frac{1}{2}\sin\left(\theta_1(t)-\theta_2(t)\right.\\
&\quad \left.-\theta_3(t)\right), \sin\left(\theta_1(t)\right), 0.1500\cos\left(\theta_1(t)\right)+0.1905\cos\left(\theta_1(t)\right.\\
&\quad \left.-\theta_2(t)-\theta_3(t)\right)+0.1905\cos\left(\theta_1(t)+\theta_2(t)+\theta_3(t)\right)\\
&\quad \left.+0.1905\cos\left(\theta_1(t)-\theta_2(t)\right)+0.1905\cos\left(\theta_1(t)+\theta_2(t)\right)\right],\\
&\quad \left[\frac{1}{2}\sin\left(\theta_1(t)+\theta_2(t)+\theta_3(t)\right)+\frac{1}{2}\sin\left(\theta_1(t)-\theta_2(t)\right.\right.\\
&\quad \left.-\theta_3(t)\right), -\frac{1}{2}\cos\left(\theta_1(t)-\theta_2(t)-\theta_3(t)\right)+\frac{1}{2}\cos\left(\theta_1(t)\right.\\
&\quad \left.+\theta_2(t)+\theta_3(t)\right), -\cos\left(\theta_1(t)\right), 0.1500\sin\left(\theta_1(t)\right)\\
&\quad +0.1905\sin\left(\theta_1(t)+\theta_2(t)+\theta_3(t)\right)+0.1905\sin\left(\theta_1(t)\right.\\
&\quad \left.-\theta_2(t)-\theta_3(t)\right)+0.1905\sin\left(\theta_1(t)+\theta_2(t)\right)\\
&\quad \left.+0.1905\sin\left(\theta_1(t)-\theta_2(t)\right)\right],\\
&\quad \left[\sin\left(\theta_2(t)+\theta_3(t)\right), \cos\left(\theta_2(t)+\theta_3(t)\right), 0., 0.3810\sin\left(\theta_2(t)\right.\right.\\
&\quad \left.\left.+\theta_3(t)\right)+0.06400+0.3810\sin\left(\theta_2(t)\right)\right],\\
&\quad \left.\left[0., 0., 0., 1.\right]\right]
\end{aligned}
$$

$C03d := map(diff, T03, t) :$

EXTRACTING x y and z

```
C03dx := C03d[1, 4] :

C03dy := C03d[2, 4] :

C03dz := C03d[3, 4] :

C03xd1 := diff (C03dx, t) :

C03yd1 := diff (C03dy, t) :

C03zd1 := diff (C03dz, t) :
```

COG-T04

```
C04 := combine(simplify(multiply(T01, T125, T235, T345))) :

C04d := map(diff, T04, t) :
```

EXTRACTING x y and z

```
C04dx := C04d[1, 4] :

C04dy := C04d[2, 4] :

C04dz := C04d[3, 4] :

 C04xd1 := diff (C04dx, t) :

C04yd1 := diff (C04dy, t) :

C04zd1 := diff (C04dz, t) :
```

T04

```
T04 := combine(simplify(multiply(T01, T12, T23, T34))) :

T04d := map(diff, T04, t) :
```

EXTRACTING x y and z

```
T04dx := T04d[1, 4] :

T04dy := T04d[2, 4] :
```

$T04dz := T04d[3,4]:$

$xd1 := diff(T04dx, t):$

$yd1 := diff(T04dy, t):$

$zd1 := diff(T04dz, t):$

Lagrangian Energy

KINETIC ENERGY

$K1 := \left(1/2 * m_1 * \left(xd1^2 + yd1^2 + zd1^2\right)\right):$

POTENTIAL ENERGY

$$P1 := \left(\frac{1}{2} \cdot m_1 \cdot g \cdot (T04dx - C02dx)\right) + \left(\frac{1}{2} \cdot m_1 \cdot g \cdot (T04dx - C03dx)\right) + \left(\frac{1}{2} \cdot m_1 \cdot g \cdot (T04dx - C04dx)\right):$$

Lagrangian Solution

$L := K1 - P1$

Substitutions

Substitute in temporary variables and take partial derivatives wrt theta dot

$$L1 := subs\left(\left\{\theta_1(t) = T1, diff\left(\theta_1(t), t\right) = dT1, \theta_2(t) = T2, diff\left(\theta_2(t), t\right) = dT2, \theta_3(t) = T3, diff\left(\theta_3(t), t\right) = dT3, \theta_4(t) = T4, diff\left(\theta_4(t), t\right) = dT4\right\}, L\right);$$

$$
\begin{aligned}
&\frac{1}{2}\, m_1 \Bigg(\Bigg(-\frac{1}{2}\, l_3 \cos(\mathit{T1}-\mathit{T3}-\mathit{T4}-\mathit{T2})\, (\mathit{dT1}-\mathit{dT3}-\mathit{dT4} \\
&\quad -\mathit{dT2})^2 - \frac{1}{2}\, l_3 \sin(\mathit{T1}-\mathit{T3}-\mathit{T4}-\mathit{T2}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} \\
&\quad - \Big(\frac{\partial}{\partial t}\, \mathit{dT3} \Big) - \Big(\frac{\partial}{\partial t}\, \mathit{dT4} \Big) - \Big(\frac{\partial}{\partial t}\, \mathit{dT2} \Big) \Big) - \frac{1}{2}\, l_3 \cos(\mathit{T1} \\
&\quad + \mathit{T3}+\mathit{T4}+\mathit{T2})\, (\mathit{dT1}+\mathit{dT3}+\mathit{dT4}+\mathit{dT2})^2 - \frac{1}{2}\, l_3 \sin(\mathit{T1} \\
&\quad + \mathit{T3}+\mathit{T4}+\mathit{T2}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} + \frac{\partial}{\partial t}\, \mathit{dT3} + \frac{\partial}{\partial t}\, \mathit{dT4} + \frac{\partial}{\partial t}\, \mathit{dT2} \Big) \\
&\quad - \frac{1}{2}\, l_2 \cos(\mathit{T1}-\mathit{T2}-\mathit{T3})\, (\mathit{dT1}-\mathit{dT2}-\mathit{dT3})^2 \\
&\quad - \frac{1}{2}\, l_2 \sin(\mathit{T1}-\mathit{T2}-\mathit{T3}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} - \Big(\frac{\partial}{\partial t}\, \mathit{dT2} \Big) \\
&\quad - \Big(\frac{\partial}{\partial t}\, \mathit{dT3} \Big) \Big) - \frac{1}{2}\, l_2 \cos(\mathit{T1}+\mathit{T2}+\mathit{T3})\, (\mathit{dT1}+\mathit{dT2} \\
&\quad + \mathit{dT3})^2 - \frac{1}{2}\, l_2 \sin(\mathit{T1}+\mathit{T2}+\mathit{T3}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} + \frac{\partial}{\partial t}\, \mathit{dT2} \\
&\quad + \frac{\partial}{\partial t}\, \mathit{dT3} \Big) - \frac{1}{2}\, l_1 \cos(\mathit{T1}-\mathit{T2})\, (\mathit{dT1}-\mathit{dT2})^2 \\
&\quad - \frac{1}{2}\, l_1 \sin(\mathit{T1}-\mathit{T2}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} - \Big(\frac{\partial}{\partial t}\, \mathit{dT2} \Big) \Big) - \frac{1}{2}\, l_1 \cos(\mathit{T1} \\
&\quad + \mathit{T2})\, (\mathit{dT1}+\mathit{dT2})^2 - \frac{1}{2}\, l_1 \sin(\mathit{T1}+\mathit{T2}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} \\
&\quad + \frac{\partial}{\partial t}\, \mathit{dT2} \Big) - \mathit{offset1}\, \cos(\mathit{T1})\, \mathit{dT1}^2 \\
&\quad - \mathit{offset1}\, \sin(\mathit{T1}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} \Big) \Bigg)^2 + \Bigg(-\frac{1}{2}\, l_3 \sin(\mathit{T1}+\mathit{T3} \\
&\quad + \mathit{T4}+\mathit{T2})\, (\mathit{dT1}+\mathit{dT3}+\mathit{dT4}+\mathit{dT2})^2 + \frac{1}{2}\, l_3 \cos(\mathit{T1}+\mathit{T3} \\
&\quad + \mathit{T4}+\mathit{T2}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} + \frac{\partial}{\partial t}\, \mathit{dT3} + \frac{\partial}{\partial t}\, \mathit{dT4} + \frac{\partial}{\partial t}\, \mathit{dT2} \Big) \\
&\quad - \frac{1}{2}\, l_3 \sin(\mathit{T1}-\mathit{T3}-\mathit{T4}-\mathit{T2})\, (\mathit{dT1}-\mathit{dT3}-\mathit{dT4}-\mathit{dT2})^2 \\
&\quad + \frac{1}{2}\, l_3 \cos(\mathit{T1}-\mathit{T3}-\mathit{T4}-\mathit{T2}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} - \Big(\frac{\partial}{\partial t}\, \mathit{dT3} \Big) \\
&\quad - \Big(\frac{\partial}{\partial t}\, \mathit{dT4} \Big) - \Big(\frac{\partial}{\partial t}\, \mathit{dT2} \Big) \Big) - \frac{1}{2}\, l_2 \sin(\mathit{T1}+\mathit{T2} \\
&\quad + \mathit{T3})\, (\mathit{dT1}+\mathit{dT2}+\mathit{dT3})^2 + \frac{1}{2}\, l_2 \cos(\mathit{T1}+\mathit{T2} \\
&\quad + \mathit{T3}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} + \frac{\partial}{\partial t}\, \mathit{dT2} + \frac{\partial}{\partial t}\, \mathit{dT3} \Big) - \frac{1}{2}\, l_2 \sin(\mathit{T1}-\mathit{T2} \\
&\quad - \mathit{T3})\, (\mathit{dT1}-\mathit{dT2}-\mathit{dT3})^2 + \frac{1}{2}\, l_2 \cos(\mathit{T1}-\mathit{T2} \\
&\quad - \mathit{T3}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} - \Big(\frac{\partial}{\partial t}\, \mathit{dT2} \Big) - \Big(\frac{\partial}{\partial t}\, \mathit{dT3} \Big) \Big) - \frac{1}{2}\, l_1 \sin(\mathit{T1} \\
&\quad + \mathit{T2})\, (\mathit{dT1}+\mathit{dT2})^2 + \frac{1}{2}\, l_1 \cos(\mathit{T1}+\mathit{T2}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} \\
&\quad + \frac{\partial}{\partial t}\, \mathit{dT2} \Big) - \frac{1}{2}\, l_1 \sin(\mathit{T1}-\mathit{T2})\, (\mathit{dT1}-\mathit{dT2})^2 \\
&\quad + \frac{1}{2}\, l_1 \cos(\mathit{T1}-\mathit{T2}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} - \Big(\frac{\partial}{\partial t}\, \mathit{dT2} \Big) \Big) \\
&\quad - \mathit{offset1}\, \sin(\mathit{T1})\, \mathit{dT1}^2 + \mathit{offset1}\, \cos(\mathit{T1}) \Big(\frac{\partial}{\partial t}\, \mathit{dT1} \Big) \Bigg)^2 \\
&\quad + \Bigg(-l_3 \sin(\mathit{T3}+\mathit{T4}+\mathit{T2})\, (\mathit{dT3}+\mathit{dT4}+\mathit{dT2})^2 + l_3 \cos(\mathit{T3} \\
&\quad + \mathit{T4}+\mathit{T2}) \Big(\frac{\partial}{\partial t}\, \mathit{dT3} + \frac{\partial}{\partial t}\, \mathit{dT4} + \frac{\partial}{\partial t}\, \mathit{dT2} \Big) - l_2 \sin(\mathit{T2} \\
&\quad + \mathit{T3})\, (\mathit{dT2}+\mathit{dT3})^2 + l_2 \cos(\mathit{T2}+\mathit{T3}) \Big(\frac{\partial}{\partial t}\, \mathit{dT2} + \frac{\partial}{\partial t}\, \mathit{dT3} \Big) \\
&\quad - l_1 \sin(\mathit{T2})\, \mathit{dT2}^2 + l_1 \cos(\mathit{T2}) \Big(\frac{\partial}{\partial t}\, \mathit{dT2} \Big) \Bigg)^2 \Bigg) - 14.72\, m_1 \Bigg(\\
&\quad -\frac{1}{2}\, l_3 \sin(\mathit{T1}-\mathit{T3}-\mathit{T4}-\mathit{T2})\, (\mathit{dT1}-\mathit{dT3}-\mathit{dT4}-\mathit{dT2}) \\
&\quad - \frac{1}{2}\, l_3 \sin(\mathit{T1}+\mathit{T3}+\mathit{T4}+\mathit{T2})\, (\mathit{dT1}+\mathit{dT3}+\mathit{dT4}+\mathit{dT2}) \\
&\quad - \frac{1}{2}\, l_2 \sin(\mathit{T1}-\mathit{T2}-\mathit{T3})\, (\mathit{dT1}-\mathit{dT2}-\mathit{dT3}) - \frac{1}{2}\, l_2 \sin(\mathit{T1} \\
&\quad + \mathit{T2}+\mathit{T3})\, (\mathit{dT1}+\mathit{dT2}+\mathit{dT3}) - \frac{1}{2}\, l_1 \sin(\mathit{T1}-\mathit{T2})\, (\mathit{dT1} \\
&\quad - \mathit{dT2}) - \frac{1}{2}\, l_1 \sin(\mathit{T1}+\mathit{T2})\, (\mathit{dT1}+\mathit{dT2}) \\
&\quad - \mathit{offset1}\, \sin(\mathit{T1})\, \mathit{dT1} \Bigg)
\end{aligned}
$$

$$\mathit{Expr1} := \mathit{diff}\,(\mathit{L1}, \mathit{dT1})\ :$$

$$\mathit{Expr2} := \mathit{diff}\,(\mathit{L1}, \mathit{dT2})\ :$$

$$\mathit{Expr3} := \mathit{diff}\,(\mathit{L1}, \mathit{dT3})\ :$$

$$\mathit{Expr4} := \mathit{diff}\,(\mathit{L1}, \mathit{dT4})\ :$$

Substitute back original variables and take derivatives wrt time

$$\mathit{Expr5} := \mathit{subs}\Big(\Big\{\mathit{T1} = \theta_1(t), \mathit{dT1} = \mathit{diff}\left(\theta_1(t), t\right), \mathit{T2} = \theta_2(t), \mathit{dT2} = \mathit{diff}\left(\theta_2(t), t\right), \mathit{T3} = \theta_3(t), \mathit{dT3} = \mathit{diff}\left(\theta_3(t), t\right), \mathit{T4} = \theta_4(t), \mathit{dT4} = \mathit{diff}\left(\theta_4(t), t\right)\Big\}, \mathit{Expr1}\Big)\ :$$

$$\mathit{Expr6} := \mathit{subs}\Big(\Big\{\mathit{T1} = \theta_1(t), \mathit{dT1} = \mathit{diff}\left(\theta_1(t), t\right), \mathit{T2} = \theta_2(t), \mathit{dT2} = \mathit{diff}\left(\theta_2(t), t\right), \mathit{T3} = \theta_3(t), \mathit{dT3} = \mathit{diff}\left(\theta_3(t), t\right), \mathit{T4} = \theta_4(t), \mathit{dT4} = \mathit{diff}\left(\theta_4(t), t\right)\Big\}, \mathit{Expr2}\Big)\ :$$

$$\mathit{Expr7} := \mathit{subs}\Big(\Big\{\mathit{T1} = \theta_1(t), \mathit{dT1} = \mathit{diff}\left(\theta_1(t), t\right), \mathit{T2} = \theta_2(t), \mathit{dT2} = \mathit{diff}\left(\theta_2(t), t\right), \mathit{T3} = \theta_3(t), \mathit{dT3} = \mathit{diff}\left(\theta_3(t), t\right), \mathit{T4} = \theta_4(t), \mathit{dT4} = \mathit{diff}\left(\theta_4(t), t\right)\Big\}, \mathit{Expr3}\Big)\ :$$

$$\mathit{Expr8} := \mathit{subs}\Big(\Big\{\mathit{T1} = \theta_1(t), \mathit{dT1} = \mathit{diff}\left(\theta_1(t), t\right), \mathit{T2} = \theta_2(t), \mathit{dT2} = \mathit{diff}\left(\theta_2(t), t\right), \mathit{T3} = \theta_3(t), \mathit{dT3} = \mathit{diff}\left(\theta_3(t), t\right), \mathit{T4} = \theta_4(t), \mathit{dT4} = \mathit{diff}\left(\theta_4(t), t\right)\Big\}, \mathit{Expr4}\Big)\ :$$

$$\mathit{Expr9} := \mathit{diff}\,(\mathit{Expr5}, t)\ :$$

$$\mathit{Expr10} := \mathit{diff}\,(\mathit{Expr6}, t)\ :$$

$$\mathit{Expr11} := \mathit{diff}\,(\mathit{Expr7}, t)\ :$$

$$\mathit{Expr12} := \mathit{diff}\,(\mathit{Expr8}, t)\ :$$

Take partial derivatives wrt theta and substitute back original variables

$$\mathit{Expr13} := \mathit{diff}\,(\mathit{L1}, \mathit{T1})\ :$$

$$\mathit{Expr14} := \mathit{diff}\,(\mathit{L1}, \mathit{T2})\ :$$

$$\mathit{Expr15} := \mathit{diff}\,(\mathit{L1}, \mathit{T3})\ :$$

$\mathit{Expr16} := \mathit{diff}\,(\mathit{L1}, \mathit{T4})\ :$

$$\begin{aligned}&\mathit{Expr17} := \mathit{subs}\Big(\Big\{\mathit{T1} = \theta_1(t), \mathit{dT1} = \mathit{diff}\,\big(\theta_1(t), t\big), \mathit{T2} = \theta_2(t), \mathit{dT2}\\ &\quad = \mathit{diff}\,\big(\theta_2(t), t\big), \mathit{T3} = \theta_3(t), \mathit{dT3} = \mathit{diff}\,\big(\theta_3(t), t\big), \mathit{T4} = \theta_4(t), \mathit{dT4}\\ &\quad = \mathit{diff}\,\big(\theta_4(t), t\big)\Big\}, \mathit{Expr13}\Big)\ :\end{aligned}$$

$$\begin{aligned}&\mathit{Expr18} := \mathit{subs}\Big(\Big\{\mathit{T1} = \theta_1(t), \mathit{dT1} = \mathit{diff}\,\big(\theta_1(t), t\big), \mathit{T2} = \theta_2(t), \mathit{dT2}\\ &\quad = \mathit{diff}\,\big(\theta_2(t), t\big), \mathit{T3} = \theta_3(t), \mathit{dT3} = \mathit{diff}\,\big(\theta_3(t), t\big), \mathit{T4} = \theta_4(t), \mathit{dT4}\\ &\quad = \mathit{diff}\,\big(\theta_4(t), t\big)\Big\}, \mathit{Expr14}\Big)\ :\end{aligned}$$

$$\begin{aligned}&\mathit{Expr19} := \mathit{subs}\Big(\Big\{\mathit{T1} = \theta_1(t), \mathit{dT1} = \mathit{diff}\,\big(\theta_1(t), t\big), \mathit{T2} = \theta_2(t), \mathit{dT2}\\ &\quad = \mathit{diff}\,\big(\theta_2(t), t\big), \mathit{T3} = \theta_3(t), \mathit{dT3} = \mathit{diff}\,\big(\theta_3(t), t\big), \mathit{T4} = \theta_4(t), \mathit{dT4}\\ &\quad = \mathit{diff}\,\big(\theta_4(t), t\big)\Big\}, \mathit{Expr15}\Big)\ :\end{aligned}$$

$$\begin{aligned}&\mathit{Expr20} := \mathit{subs}\Big(\Big\{\mathit{T1} = \theta_1(t), \mathit{dT1} = \mathit{diff}\,\big(\theta_1(t), t\big), \mathit{T2} = \theta_2(t), \mathit{dT2}\\ &\quad = \mathit{diff}\,\big(\theta_2(t), t\big), \mathit{T3} = \theta_3(t), \mathit{dT3} = \mathit{diff}\,\big(\theta_3(t), t\big), \mathit{T4} = \theta_4(t), \mathit{dT4}\\ &\quad = \mathit{diff}\,\big(\theta_4(t), t\big)\Big\}, \mathit{Expr16}\Big)\ :\end{aligned}$$

Solve for the Lagrange Equations!

$\mathit{T1} := \mathit{Expr5} - \mathit{Expr17};$

$\mathit{T2} := \mathit{Expr6} - \mathit{Expr18};$

$\mathit{T3} := \mathit{Expr7} - \mathit{Expr19};$

$\mathit{T4} := \mathit{Expr8} - \mathit{Expr20};$

APPENDIX B: COMMUNICATION API

Sabertooth SBRIO UDP API

Control Terminology

Each joint will have an ID associated with it. The ID will consist of a Leg indicator as well as a joint number. Possible joint IDs are listed below.

Joint ID	Description
FL1	Front Left Hip
FL2	Front Left Shoulder
FL3	Front Left Knee
FR1	Front Right Hip
FR2	Front Right Shoulder
FR3	Front Right Knee
RL1	Rear Left Hip
RL2	Rear Left Shoulder
RL3	Rear Left Knee
RR1	Rear Right Hip
RR2	Rear Right Shoulder
RR3	Rear Right Knee

Computer -> SBRIO Packet

This packet contains all of the control for the robot and is sent on 50ms intervals. After this packet is received the data will be latched to registers on the SBRIO and a return packet will be sent to

Computer. The header for this packet is 0x00 and this will inform the SBRIO that it is a control packet.

Each joint requires both a Position and Velocity in order to properly control the PID Motor Controller. Both the position and velocity values will be checked against a range (Internal to FPGA) and will return error state.

Position Ranges will depend on the joint be controlled, all of the hips will have a range of approximately 20 degrees based on ± 10degrees setting and the other joints will have a range of 160 degrees based on a ± 80 degrees setting. These values will be controlled by two bit hex values (1 Byte Transfer).

Packet 1:6 (These are simply parts of a packet (ex: this is first 6th of packet))

0x00	FL1 Pos	FL1 Vel	FL2 Pos	FL2 Vel	FL3 Vel	FL3 Pos

Example Values

	-80(10) Degrees	0 Degrees	+80(10) Degrees
FL1 Pos	0x00	0x50	0xA0
	Full Reverse (0)	Stop (125)	Full Forward (255)
FL1 Vel	0x00	0x7D	0xFF

Packet 2:6

FR1 Pos	FR1 Vel	FR2 Pos	FR2 Vel	FR3 Pos	FR3 Vel

Packet 3:6

RL1 Pos	RL1 Vel	RL2 Pos	RL2 Vel	RL3 Pos	RL3 Vel

Packet 4:6

RR1 Pos	RR1 Vel	RR2 Pos	RR2 Vel	RR3 Pos	RR3 Vel

The control packet is also responsible for controlling the servos on the Camera Pan Tilt as well as the LEDs that are located on the robot. The Servo Controls will take in the angle the servo should go to, that being on a 180 degree range.

Packet 5:6

CAM Pitch	CAM Yaw	LEDL	LEDR

Packer 6:6

LED1	LED2	LED3	LED4

Example Values

	0 Degrees	80 Degrees
CAM Pitch & Cam Yaw	0x00	0x50
	On Off	Function
LEDn	0x01 0x00	0xnn

Calibration Packet

This packet contains all of the calibration data for the robot and is sent on 50ms intervals. After this packet is received the data will be latched to registers on the SBRIO and a return packet will be sent to Computer. The header for this packet is 0x01 and this will inform the SBRIO that it is a control packet.

Each joint will be required to be calibrated once in order to set the center angles for the joint.

For Hip Joints:

In order for a Hip to be calibrated it needs to be set to calibrate by the calibration packet. Once set to calibrate the SBRIO will start communicated with the EPOS2. This EPOS will be commanded to home to the negative position sensor. Once the maxon has homed the encoder will be set to zero, then the SBRIO will tell the EPOS2 to drive in the Positive position until it triggers the positive position sensor. When the positive position sensor is triggered the encoder count will be set to the FPGA as zero point. This value will be the exact center between the two position limit switches. This value should be accurate, however if there is any error the offset value can be used to adjust for perfect center.

For Other Joints:

The Other joints will be calibrated via a two part process, the first step will be the mechanical calibration of the potentiometers in regards to angles and the second step will be the calibration of the PID gain values. Both of these calibrations will require the use of an external computing device running labview to communicate to the FPGA modules. The mechanical calibration will utilize the FPGA VI which will be able to show the values of the Potentiometers on a computer screen. The leg will be delinked from the potentiometer and moved to the directly down position. Once in this position the potentiometers will be rotated to “center” for them (middle voltage reading). This will then have the shaft tightened on it. This will allow the potentiometers range to act as the range for the leg, providing a safe 160 degrees of motion with accurate feedback loop.

The second step will be the calibration of the PID loop. However, before calibrating the PID loop it is important to ensure that the leg has set limits within the code it knows not drive the motors past. This will help ensure PID tuning does not damage the system.   The computer will be used to monitor the position as well as the PID gain values. By adjusting the values slowly an accurate control loop will be established.

Packet 1:2

0x01	**FL1 CAL**	**FR1 CAL**	**RL1 CAL**	**RR1 CAL**

The calibration constants which are written to FL1 CAL…RR1 CAL will consist of on or off (0x01 0x00) in order to set the calibration of the hips. The FBRIO will only run the calibration series if it receives a CAL command and it is not already calibrated.

Packet2:2

FL1 OFFSET	**FR1 OFFSET**	**RL1 OFFSET**	**RR1 OFFSET**

The offset values are sent to ensure that the hip motors are properly calibrate by their internal software on the EPOS2. These values will be read by the FPGA and added/ subtracted from the encoder value that is set as the zero point.

SBRIO -> Comp Packet

In order for the onboard processor to make accurate decisions it is necessary for the sensor data on the robot to be transferred to the host computer. This sensor data will be transferred every 50ms after a packet is received. The sensor data will be the processed on the FPGA, that being the data sent to the computer will not be raw data. The data included in the packet will be joint angles, current sensors, spring force, body joint and zero positions for hip. Each of the sub packets will be a 2 digit hex 0x00.

Packet 1:6

FLP1	**FLP2**	**FLP3**	**FRP1**	**FRP2**	**FRP3**

Packet 2:6

RLP1	**RLP2**	**RLP3**	**RRP1**	**RRP2**	**RRP3**

Packet 3:6

CSFL1	**CSFL2**	**CSFL3**	**CSFR1**	**CSFR2**	**CSFR3**

Packet 4:6

CSRL1	**CSRL2**	**CSRL3**	**CSRR1**	**CSRR2**	**CSRR3**

Packet 5:6

FLS2	**FLS3**	**FRS2**	**FRS3**	**RLS2**	**RLS3**	**RRS2**	**RRS3**

Packet 6:6

BJ1	**BJ2**	**FL1 ZERO**	**FR1 ZERO**	**RL1 ZERO**	**RR1 ZERO**	**ERROR**

APPENDIX C: TEAM STATEMENT OF WORK

Sabertooth Robotics Statement of Work

Background

Sabertooth Robotics is a senior design team composed of five engineering students from the Worcester Polytechnic Institute. The team is working on an innovative robotics project whose scope is to design a high mobility, quadruped robot capable of traversing terrain otherwise impassable by wheeled vehicles. Specifically, the robot will be able to ascend and descend straight stairs with a predetermined gait. The robot will have at least an 8 inch by 12 inch stride to account for the rise and run of a staircase. With each step, the robot will be able to recover power exerted against the ground using a spring system. The robot will be able to carry a payload of 30 pounds.

This 4'x4'x3' four legged robot will be freestanding and weigh approximately 250 pounds, including a payload. The size is to accommodate for a target speed of 5 feet per second, while still being capable of climbing staircases. The robot will have three motors on each leg to power the three independent joints – the hip, the shoulder, and the knee. These joints will use simulated tendons in order to recover exerted against the ground. Power is consumed in legged robots when pressure is applied to a walking surface, both when trying to maintain a pose or transitioning between poses. Instead our robot will store energy in spring systems in the leg joints to be released in the next step. The robot will also contain a two degree of freedom mid-joint in order to allow flexibility going around tight corners and ascending stairs.

The robot will incorporate a computer vision system for staircase recognition, obstacle avoidance, and distance calculation. The team will use a video camera to provide feed to an on-board processor, which will implement object recognition algorithms to recognize where a staircase is relative to the robot. Once the position of the staircase is known, a LIDAR sensor will be used to produce a three dimensional map of the staircase to calculate the distance to the staircase and any obstacles between the robot and the stairs. This data will be used to determine a path from the robot to the stairs. After arriving at the staircase, another map will be produced to calculate the rise and run of the staircase, to be used for determining the gait the robot will use to climb the staircase.

The development of this robot will rely on research already conducted on a number of legged robots, including but not limited to Boston Dynamic's *BigDog* and *LittleDog*, Carnegie Mellon's/Michigan State's *MABEL*, and MIT's *Spring Flamingo*[1][2][3]. The research from these robotic platforms will be useful in seeing what strategies have been used and proven to already work in this field. Additionally, the team has studied specific animal properties already found in nature and will mimic some of the characteristics that they exhibit.

Description of Work

Moving forward with this project will require much work encompassing mechanical design, manufacturing, programming and electrical implementation, biological relation to quadrupedal animals, and system testing.

Mechanical Design

In order to finalize the design of this system and create a successful project there is a need to continue and involve sub-system designs. While the body joint has already been prototyped, there is still a need to prototype other systems. The next system that must be prototyped is the leg spring system. This full-scale prototype will be made out of acrylic and MDF board in order to create a low cost prototype. The prototype will be used to test the full kinematics of the proposed leg assembly, including the kinematics of the spring systems. The leg will also serve to confirm a final test for interferences, which could not easily be tested in a Computer Aided Design program.

Manufacturing

With a complete design it is then necessary to move onto manufacturing of the sub systems. The first sub system that will be manufactured will be a single complete leg and hip assembly. Once this system is complete, it is necessary to test both the leg and hip for correspondence to design specifications. Throughout sub-system testing small adjustments may need to be made in order to fix previously unseen issues. Gait tests will need to be performed on the first leg assembly. These tests will ensure that the robot can successfully complete a walking and trotting gait. Once these tests are completed, any necessary adjustments can be made and the overall design will be iterated.

The team will then complete manufacturing the four legs needed for entire robot. Each of these legs will be tested in a similar manner as the first to ensure they can complete a walking and trotting gait. After the legs and hips are completed, the team will need to create the body joint subsystem.

The body joint subsystem will be manufactured and then tested to ensure the joint operates in accordance to the prototype.

Once each of the subsystems that will provide motion in the robot have been manufactured, there are still more systems that must be created. The first of these systems is the frame segments of the robot; these will be fabricated by bending and welding pieces of tubing to the proper specifications. The last component that must be manufactured will be the assembly of the battery packs. Each battery pack consists of eight 3.6V 2300mAh A123 cells in series for a peak voltage of 28.8V. We will have 12 of these packs in parallel for a total of 27.6Ah.The batteries will be charged external of the robot with the charges provided by Venom Racing. Once all of the subsystems have been manufactured and tested, the robot will be assembled and wired.

Programming and Electrical Design

While the manufacturing is being conducted, the team will also be working on developing the software architecture to control the robot's autonomy. This will be accomplished through implementing the sensor head to deliver information to the robot about its surrounding environment. The first step will be to develop point clouds with the LIDAR sensor, including the proper filtering and shading to create usable data. With a complete point cloud it will then be necessary to localize the robot within the virtual world that was created. The LIDAR data will also be used to allow for path planning. This will be accomplished through the use of the robot's encoders, potentiometers, IMU sensor, and visual odometry. Path planning will be accomplished by considering the end goal (either where the staircase is, or where the robot can go to look for a staircase), the dimensions of the robot (where the robot can physically walk), and the ease of the path (considering elevation and length of the path). Next, recognition of staircases will be accomplished with the OpenCV library, using the SURF algorithm to detect key points of a base staircase, and to compare those key points with points of the image the camera currently sees. All of the path planning and data collection will be processed by the onboard netbook. Once the current sensor head has been implemented, the team may consider designing a new housing for aesthetic purposes.

The software architecture will also encompass gait control. The gait control will be programmed using Labview and executed on the SBRio Board. Specific gaits will be generated using the simulations developed in MATLAB. The SBRio Board will also be the source of data collection for the potentiometers and encoders, as well as the motor controllers. The Board will communicate with the on-board netbook over an Ethernet connection.

The electrical architecture will be implemented in accordance with the design diagram. This design diagram will be fully developed by 1/22/2011.

Biology

By creating a functional robotic system, the team now has the ability to use it for the research it was designed for. This system was designed to recreate biological elements within a mechanical system in order to use some of the superior qualities that animals possess and many robot systems do not. In order to test the effectiveness of this system and provide the biological research required, a number of design aspects will be examined. These design aspects include, but are not limited to, linking the spinal movement of an animal to the body joint, recreating the ankle tendon system of animals by developing a compliant ankle, exploring the curve generation of the ankle, examining the muscle/ tendon creation of shoulder/ knee and examining the center of gravity adjustments accomplished through the gaits.

Testing

In order to verify that the overall system performs to the previously specified design requirements, a series of performance tests will be conducted. These tests include stability tests for both the feet/toes and the body joint to ensure that the robot is stable throughout a walking and trotting gait. This will initially be done with the aid of a suspended, mobile test harness. Once successful, the tests will be replicated without any harness. In addition, different spring forces will be tested within the spring system to allow for different stiffness within the system. Different stiffness springs will allow us to tune the system for better power recovery, depending on the specific gait.

Constraints

Given the nature of the project, the team may run into a number of issues which may skew the timeline. These constraints are generally due to lead time. The constraints and reasons are outlined below.

Waterjetting of Parts – Waterjetting services will be a donation by Hydrocutter. While they are able to cut parts for the project at no cost, they are still a full service shop with other client demands. Our project will not be seen as top priority and may take more time to complete than our ambitious timeline demands for. A worst caser scenario delay may be up to 3 weeks. If the delay cannot be allowed for then the team will need to do more machining on their own and less sent out elsewhere.

Material Lead Time – Stock material can be bought from a number of suppliers. The general trend of these suppliers tends to be that the faster the stock material can be supplied, the higher the cost of the material. While the team will try and order all the stock material we will need in one large order, there will likely be unforeseen stock material which will need to be additionally ordered. Any material which will need to be ordered after the first initial bulk order may cause delays in the manufacturing of the frame. This issue can be solved with an increased budget but the team will make every possible attempt to order all materials in a timely fashion.

Machining Backup – WPI maintains their CNC machines to the best of their ability. However, despite this effort, machines can break or be reserved by other students. We, as a group will work to try and work around such machining constraints but given the scope and amount of manufacturing necessary to complete the project, may run into some issues. If a problem does occur with the machines we will seek the help of outside machine shops and see if they would be willing to let us work in their area. We do not foresee this being our greatest concern at this time.

Motors not powerful enough – Through the first two terms of this MQP the team spent a great amount of time picking motors which would be powerful enough for this application. While the team has faith that the motors are strong enough, if they happen to not meet our requirements we will need to redesign some of the system. This may result in us not meeting our design specification of walking at 5 feet per second due to the need to additionally gear down the motors.

Electronics/Component burn out – The team will make the strongest attempt to accurately wire the robot and specify all the correct components in the wiring schematic. Nonetheless, the team recognizes that sometimes things may not go according to plan which may cause unforeseen delay. Since the electrical portion of our project is entirely commercial of the shelf the components we will have technical support when necessary. The team will make sure to consult with any necessary technical support before taking any unsure steps.

Acceptance Criteria

The team has developed a clearly defined set of subsystems which we believe are crucial in order to fully complete the project. First, we can analyze the mechanical project goals. While the robot is treated as a full system, there are a few clearly defined mechanical subsystems which must be effectively completed so that the project can be considered successful.

Hip Subsystem: The hip subsystem comprises one of the degrees of freedom required to make the robot turn in place. The subsystem will be driven by a linear actuator developed by the team. Success of the hip subsystem will be determined by whether or not the motor can cause the leg to camber ±10 degrees while standing on a surface. Success will be determined by whether or not the subsystem has enough force to allow a leg to complete a full 20 degrees of movement.

Body Joint Subsystem: The body joint is the center pivot of the robot connecting the front and back half. The joint will allow the robot to have a limited amount of yaw and roll movement. The body joint will be entirely passive. Success of this subsystem will be determined as follows: The system must be able to allow the robot to rotate in both the yaw and roll movement without restricting any motion otherwise possible with a rigid central link. The system must mimic the behavior of the spine of a quadrupedal animal while not causing any limitations which would otherwise not be visible with a rigid central joint. The team recognizes that few other robotic systems have implemented a similar central joint with varying levels of success. The team intends to investigate the importance of this joint as a result of the project and come up with further recommendations on how to improve it. Finding that the joint is unnecessary will not be seen as a failure but rather a conclusion determined from extensive testing.

Shoulder and Knee Subsystem: The system will be fully cable driven and must incorporate springs in order to accommodate surface irregularities of up to 1.5 inches without requiring a gait change. The spring system is designed so that power can be recovered with every step. For this phase of the project, however, we will not see this as a requirement for success but rather a design requirement which given the time the team would like to further investigate. All of the motors must be powerful and fast enough to achieve a 5ft/sec walking gait. The team would like to investigate the ability to achieve other gaits including the possibility of a gait to climb stairs as well as a running gait where all four legs are off the ground. While the team will design the robot with the intent of being able to climb stairs the first phase of the project will focus more on achieving a stable gait before shifting focus to climbing stairs.

Ankle Subsystem: This subsystem will be used to mimic the toe-to-heel walking movement found in dogs and other quadrupedal animals. This walking style is important because it has been proven to be the most power-efficient way of walking for animals. The ankle system will use a series of springs and dampeners to allow the robot to gain an extra level of compliance at the ankle joint. Similar to animals in nature, the back and front legs will be designed differently, each to mimic the respective leg of the animal. The criteria for success will be measured as follows: the ankle system must complete the toe-to-heel walking motion of quadrupeds while not suffering any additional drawbacks from simply having a rigidly attached link.

Sensor Head Subsystem: The sensor head can be viewed from two positions - the mechanical and system integration. The sensor head will be located at the front of the robot and will need to hold the camera and LIDAR unit. The camera will be mounted on two servos with a pan tilt mechanism allowing the robot to “see” in front and at least 120 degrees to either side. The LIDAR unit will be mounted on a scanning servo to allow the robot to develop point clouds of the surrounding environment. The system will prove to be successful if we are able to develop useful point clouds which will allow us to plan a path to navigate across the environment. The environment will consist of an indoor path with walls, predetermined obstacles, and a staircase.

Controls: The Sensor head will connect directly to the onboard netbook computer. The computer will run a meta-operating system called the Robotic Operating System (ROS). Success with the netbook will be determined based on the computer's ability to process all the necessary data in an appropriate amount of time. The netbook will be primarily responsible for determining navigation packages and communicating the appropriate gait algorithms to a gait controller (FPGA – National Instrument SBRio 9632/9632XT.

Responsibility Assignment Matrix

Task	**Responsible (R)**	**Assists (A)**	**Consulted (C)**	**Informed (I)**
Hip Subsystem (Mechanical)	Jeffrey O'Rourke	Vadim Chernyak	Jonathan Morgan	Anton Zalutsky
Body Joint Subsystem (Mechanical)	Vadim Chernyak	Jeffrey O'Rourke	Jonathan Morgan	Timothy Flynn
Sensor Head Subsystem (Mechanical)	Jonathan Morgan	Vadim Chernyak	Timothy Flynn	Jeffrey O'Rourke
Body Frame Design (Mechanical)	Jeffrey O'Rourke	Jonathan Morgan	Vadim Chernyak	Anton Zalutsky
Leg/Cable/Spring Design Subsystem	Vadim Chernyak	Jonathan Morgan	Jeffrey O'Rourke	Anton Zalutsky
Ankle Design	Jonathan Morgan	Jeffrey O'Rourke	Vadim Chernyak	Anton Zalutsky
Manufacturing Logistics	Vadim Chernyak	Jonathan Morgan	Jeffrey O'Rourke	Anton Zalutsky
MATLAB Simulations	Anton Zalutsky	Vadim Chernyak	Timothy Flynn	Jonathan Morgan
Sensor Head Integration	Timothy Flynn	Anton Zalutsky	Jonathan Morgan	Vadim Chernyak
SBRio Board Integration	Anton Zalutsky	Timothy Flynn	Jonathan Morgan	Jeffrey O'Rourke
ROS/Rviz Integration	Timothy Flynn	Anton Zalutsky	Vadim Chernyak	Jonathan Morgan

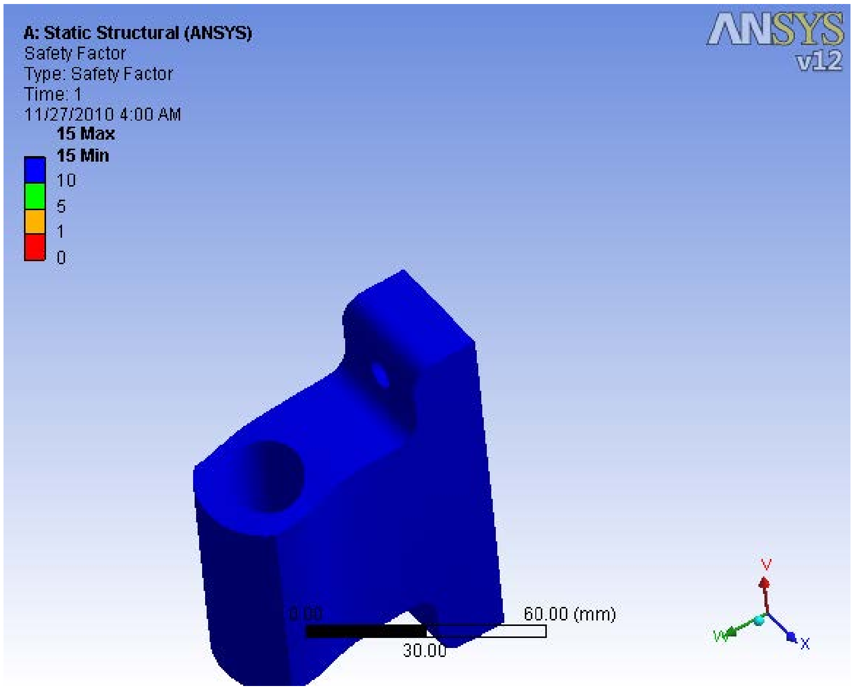
References

[1] Grizzle, J. W., Jonathan Hurst, Benhamin Morris, Hae-Won Park, and Koushil Sreenath. "MABEL, A New Robotic Bipedal Walker and Runner." (2009). University of Michigan. Web. 10 Oct. 2010. <http://www.eecs.umich.edu/~grizzle/papers/ACC_MABEL_2009.pdf>.

[2] "MIT LegLab Recent Robot Videos." *MIT Computer Science and Artificial Intelligence Laboratory | CSAIL*. Web. 09 Jan. 2011. <http://www.ai.mit.edu/projects/leglab/mpeg_vcd/>.

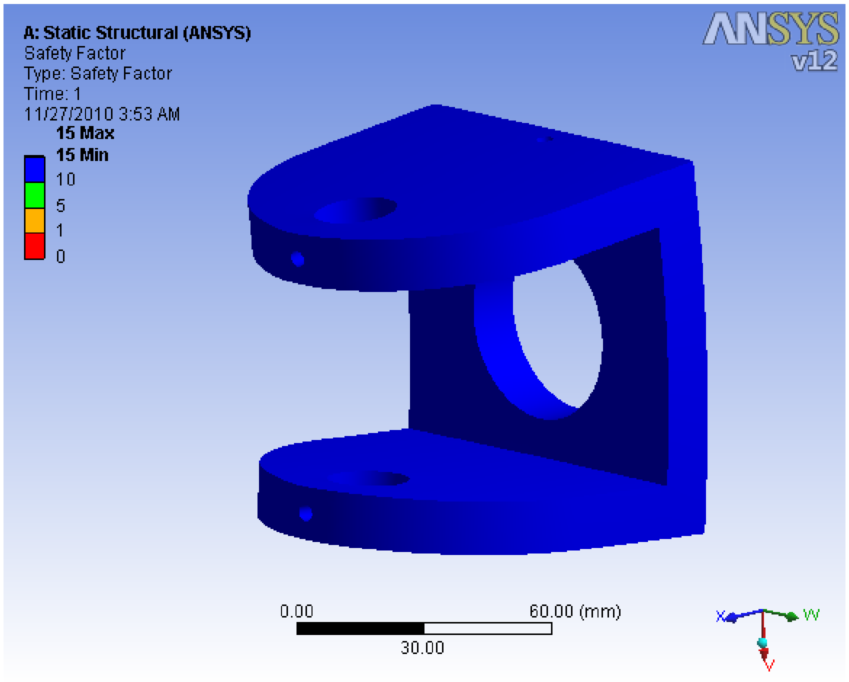
[3] Nelson, G., K. Blankespoor, and M. Raibert. "Walking BigDog: Insights and Challenges from Legged Robotics." *Journal of Biomechanics* 39 (2006): S360. Print.

APPENDIX D: FEA MODELS OF CRITICAL COMPONENTS



A: Static Structural (ANSYS)
Safety Factor
Type: Safety Factor
Time: 1
11/27/2010 4:00 AM

Figure 137- Body Joint Bushing



A: Static Structural (ANSYS)
Safety Factor
Type: Safety Factor
Time: 1
11/27/2010 3:53 AM

Figure 138- Body Mid Joint

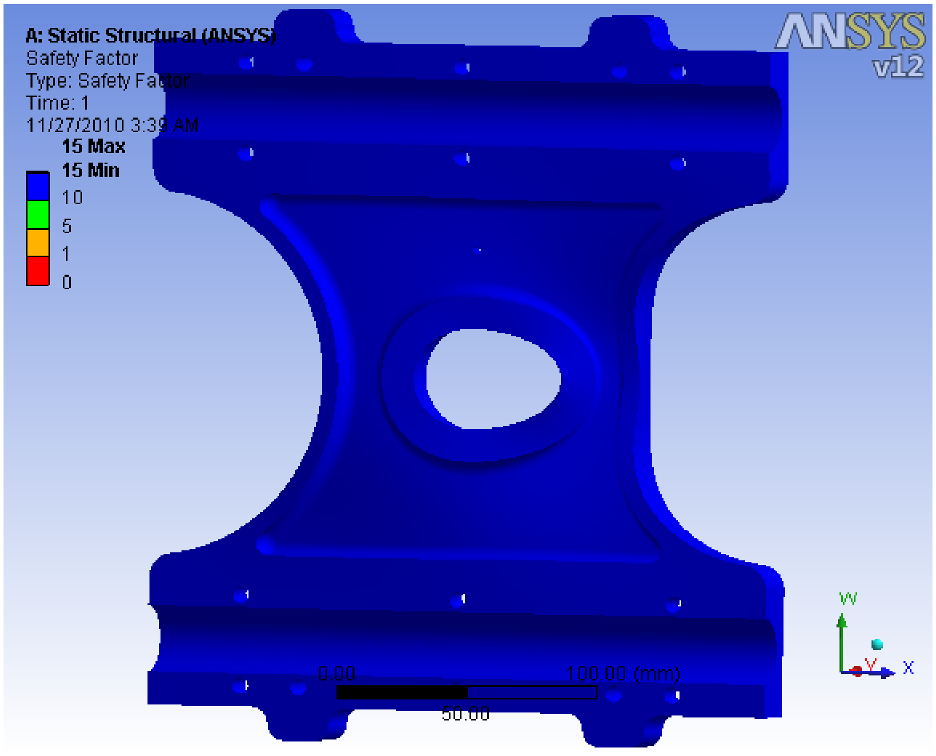


Figure 139- Body Joint Plate

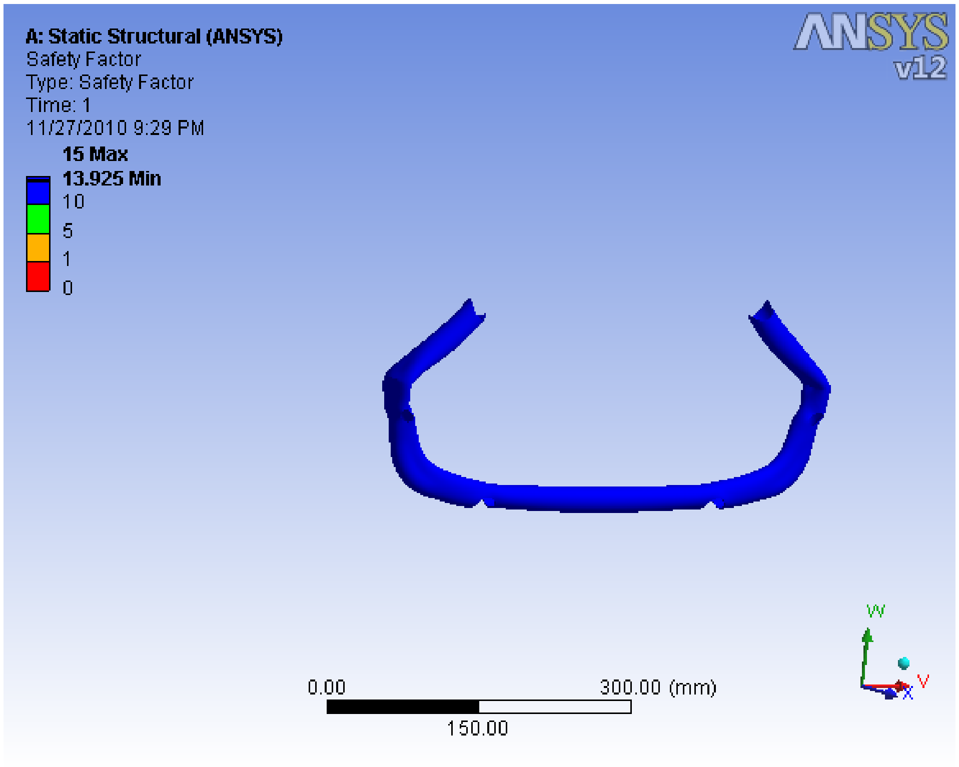


Figure 140- Hip Mount Frame Member

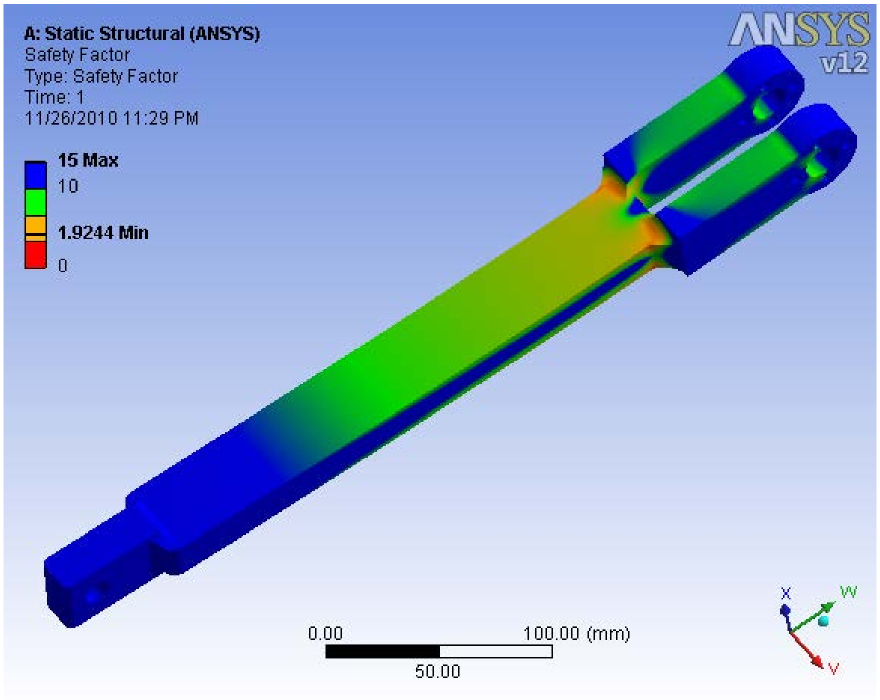


Figure 141- Lower Leg Link

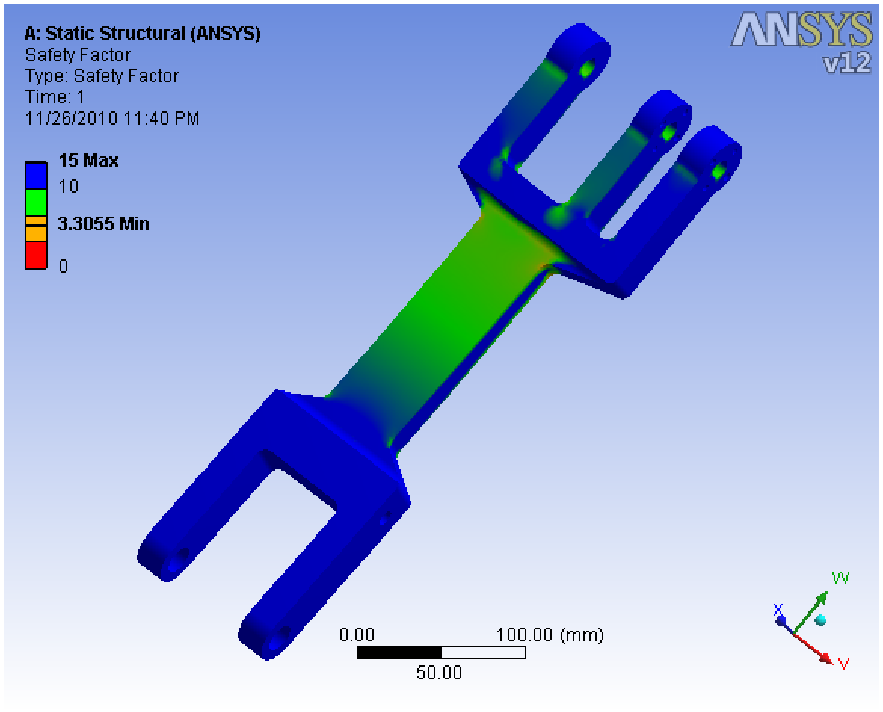


Figure 142- Upper Leg Link

APPENDIX E: ROBOT SAFETY PROCEDURE

Incident Report:

So as we all know we had an issue with the motor controllers.

What happened was we connected a leg that was wired to the robot and wanted to test it and see if we can set it to a position.

What we didn't know was the motor controllers had an issue that sometimes they don't flash the memory/parameters correctly and an isolated PWM signal was not grounded resulting in too much noise in our signal.

Because the motors were connected to the motor controllers as we turned it on, a faulty signal made the motors spasm and swing undesirably (uncrimping the spring pod and releasing the cable). Because we knew something could happen we had two people hit the emergency stop button within a fraction of a second. However, due to how powerful the motors are, this still meant a large range of motion was carried through uncontrollably and the swing was near an individual, who was NOT in direct path of the leg. Everyone was attentive to a situation such as this possibly happening, hence there were no injuries or serious mechanical damage.

The above issues mentioned above have been solved and are continuously being tested each time the robot is turned on to be sure this doesn't happen again until we are VERY sure anything modified will not cause damage to our system or anyone around it.

The procedure is as followed:

1. Labview, set all output PWM signals to 0 output. (1.45ms duty cycle)

2. Disconnect all motors from motor controllers

3. Turn on robot, and have one person monitor laptop signals and two people on estop.

4. Take small motor(having no relation with our robot) with anderson connectors and test each motor controller by plugging it in. If any crazy spasm occurs, halt system and reflash motor controller (this shouldn't ever happen again due to the way we flash them now ... and not that we need to anymore). set PWM signal to something around 150 and -150 (1.6ms and 1.3ms respectively).

5. Check all potentiometer values and makes sure they are outputting properly (move legs if you have to but seeing a fluctuation is good enough).

6. Reset all output values to 0 again.

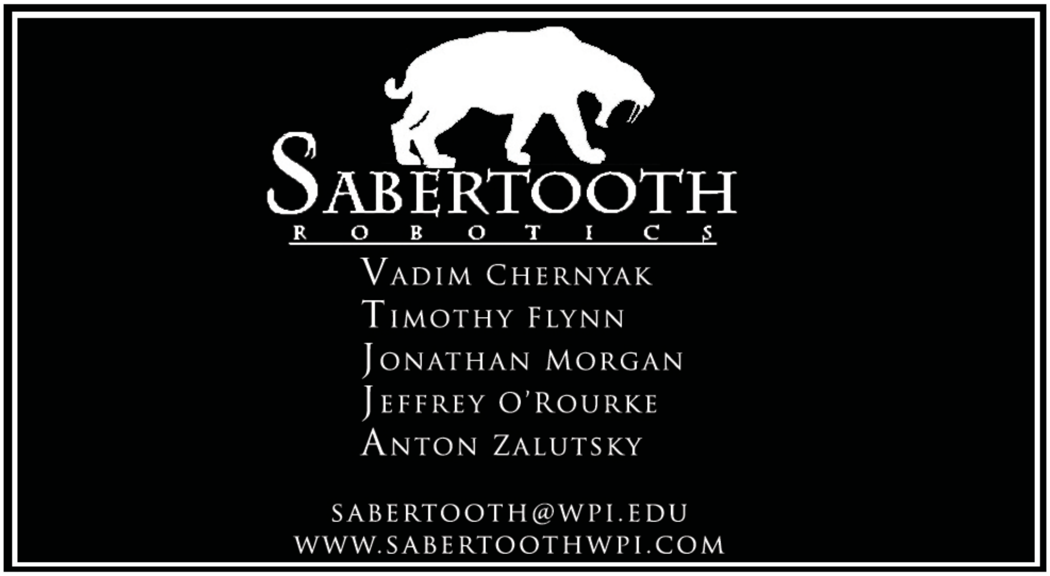
7. Plug in motors.

This method has proven to work consistently since we began implementing it.

Having two people monitor the Estop button at all time and having a software Estop ready to stop output of the motors should provide safety.

NO ONE is allowed in the path of the robot's legs path when it is on, and LED Rope is in motion to be added for extra safety to display when the legs are getting signals and when they are not. We are also planning on making an external hard e-stop, probably which will be made within the week (project presentation day is more important and we are confident currently with our safety protocol).

APPENDIX F: MARKETING MATERIALS



SABERTOOTH
ROBOTICS

VADIM CHERNYAK
TIMOTHY FLYNN
JONATHAN MORGAN
JEFFREY O'ROURKE
ANTON ZALUTSKY

SABERTOOTH@WPI.EDU
WWW.SABERTOOTHWPI.COM

Figure 143- Business Card

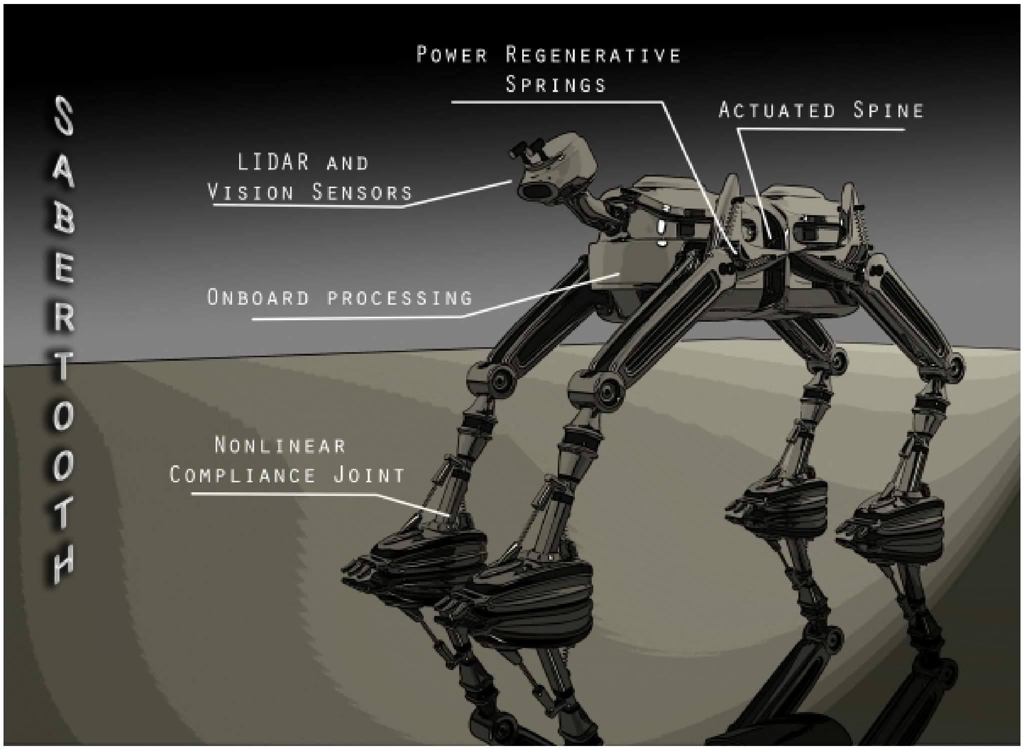
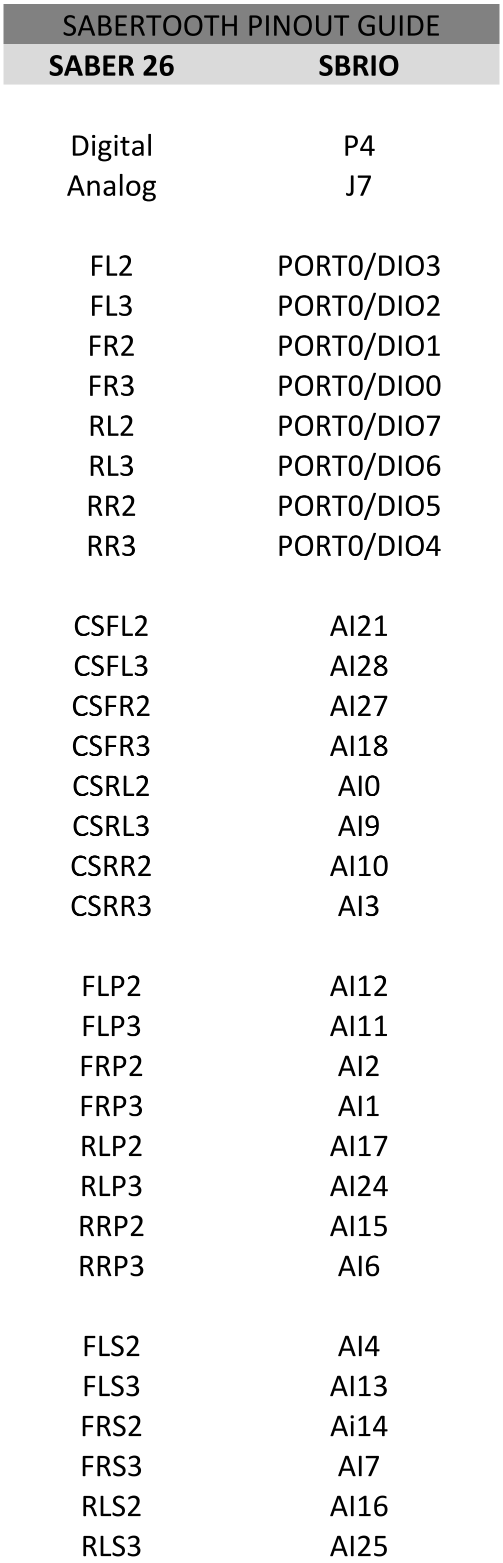


Figure 144- Concept Art

APPENDIX G: SABER BOARD PINOUT



SABERTOOTH PINOUT GUIDE	
SABER 26	**SBRIO**
Digital	P4
Analog	J7
FL2	PORT0/DIO3
FL3	PORT0/DIO2
FR2	PORT0/DIO1
FR3	PORT0/DIO0
RL2	PORT0/DIO7
RL3	PORT0/DIO6
RR2	PORT0/DIO5
RR3	PORT0/DIO4
CSFL2	AI21
CSFL3	AI28
CSFR2	AI27
CSFR3	AI18
CSRL2	AI0
CSRL3	AI9
CSRR2	AI10
CSRR3	AI3
FLP2	AI12
FLP3	AI11
FRP2	AI2
FRP3	AI1
RLP2	AI17
RLP3	AI24
RRP2	AI15
RRP3	AI6
FLS2	AI4
FLS3	AI13
FRS2	Ai14
FRS3	AI7
RLS2	AI16
RLS3	AI25

RRS2	AI26
RRS3	AI19
AUX1-D	Port2/DIO9
AUX2-D	Port1/DIO8
AUX3-D	Port1/DIO7
AUX4-D	Port1/DIO6
AUX5-A	AI30
AUX6-A	AI23
AUX7-A	AI31
AUX8-A	AI22
LED1	Port1/DIO1
LED2	Port1/DIO0
LED3	Port1/DIO9
LED4	Port0/DIO8
BJ1	AI8
BJ2	AI5
BIGLOO	NONE
LEDL	Port1/DIO3
LEDR	Port1/DIO2
YAW	Port1/DIO4
PITCH	Port1/DIO5

APPENDIX H: LEG ASSEMBLY DOCUMENT

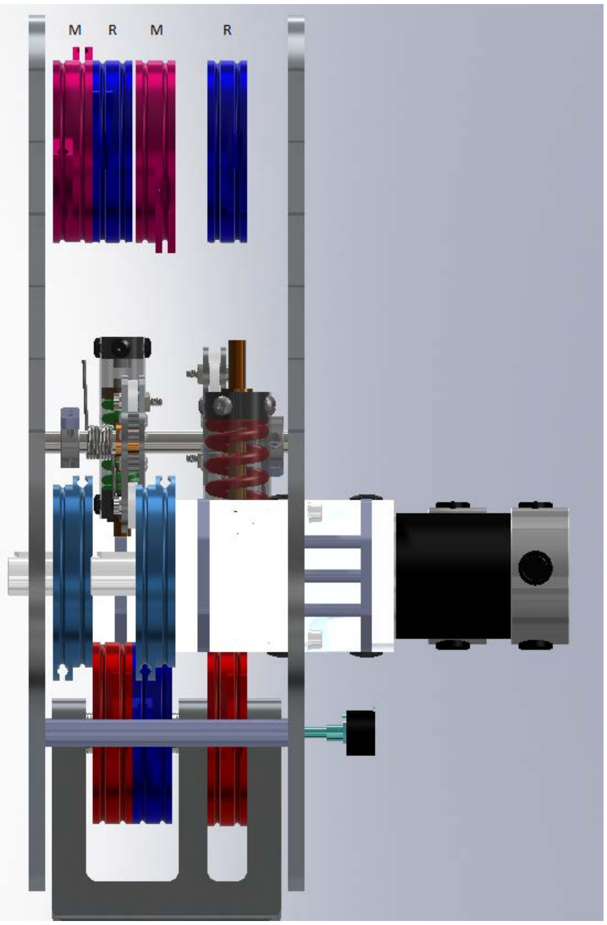
Shoulder Pulley Layout

Pink – Mirrored Pulley (M)

Blue – Regular Pulley (R)

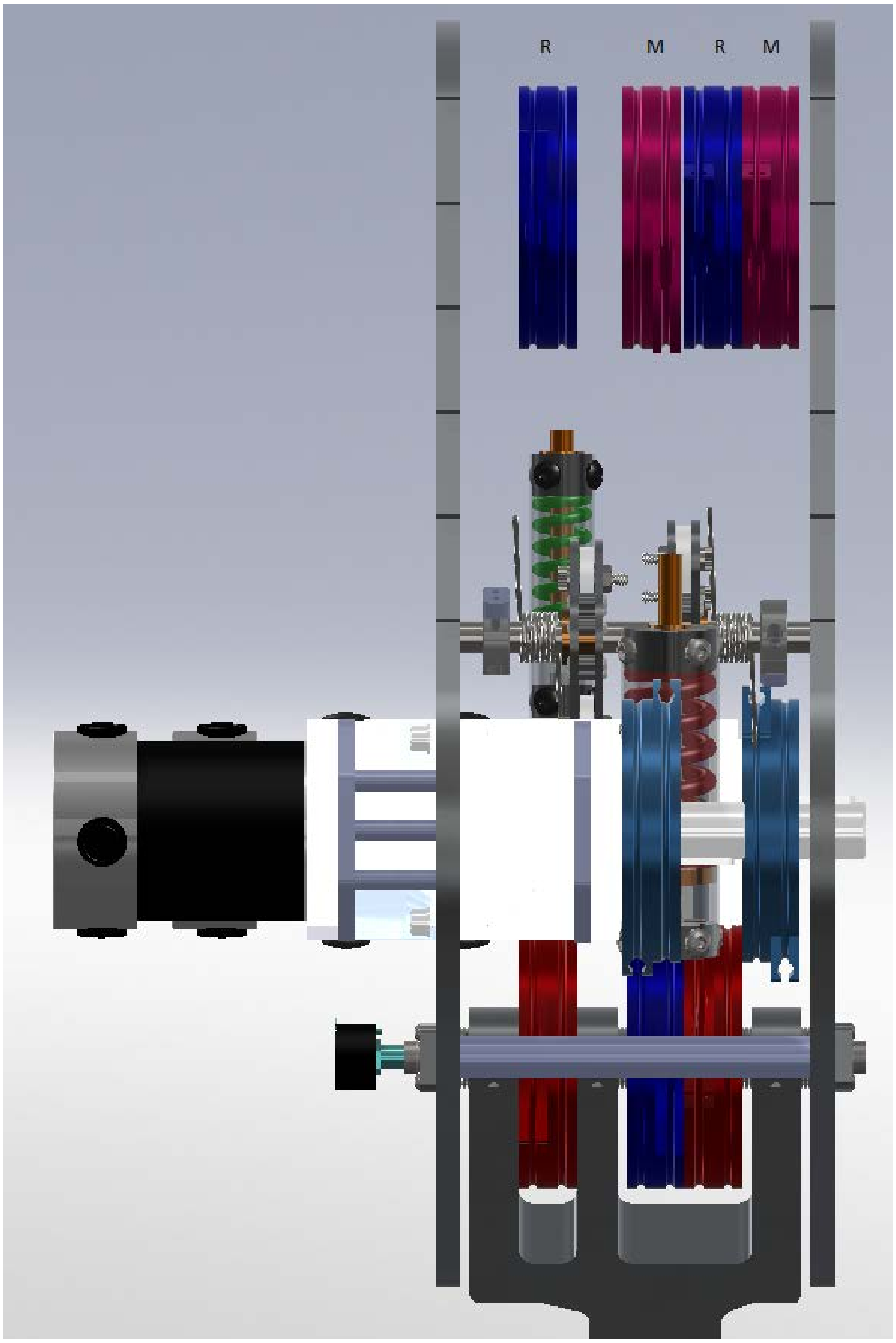
Front Left

View from Back to Front



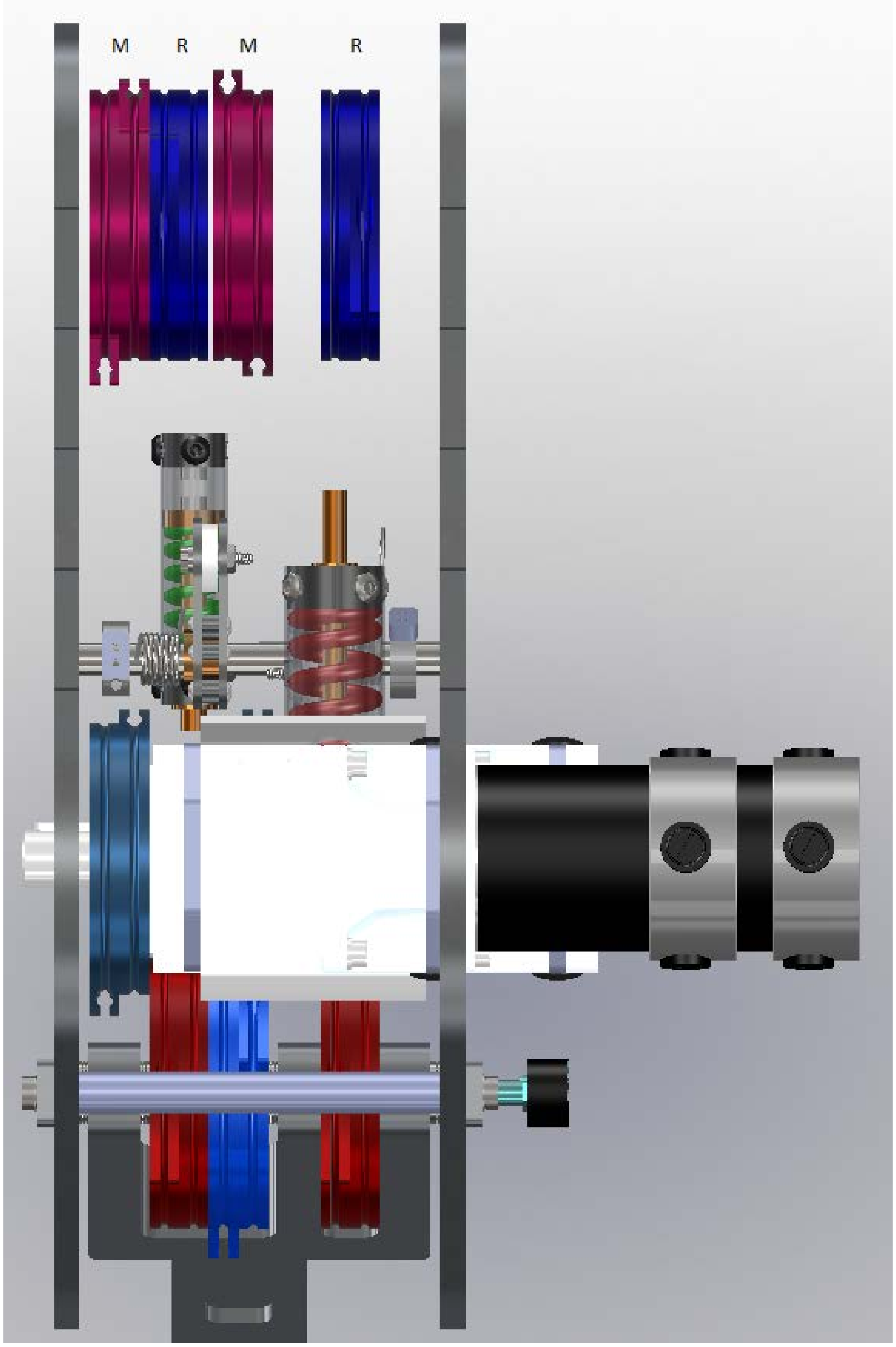
Front Right

View from Back to Front


R
M
R
M

Back Left

View from Back to Front



Back Right

View from Back To Front